ARMY RESEARCH LABORATORY

BOOM: A Computer-Aided Engineering Tool for Exterior Ballistics of Smart Projectiles

by Mark Costello and Jonathan Rogers

ARL-CR-670 **June 2011**

prepared by

Georgia Institute of Technology
Department of Aerospace Engineering

under contract

W911QX-09-C-0058

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Aberdeen Proving Ground, MD 21005

ARL-CR-670 **June 2011**

BOOM: A Computer-Aided Engineering Tool for Exterior Ballistics of Smart Projectiles

Mark Costello and Jonathan Rogers
Weapons and Materials Research Directorate, ARL

prepared by

Georgia Institute of Technology
Department of Aerospace Engineering

under contract
W911QX-09-C-0058

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
June 2011	Final	2009–2011

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
BOOM: A Computer-Aided Engineering Tool for Exterior Ballistics of Smart Projectiles	W911QX-09-C-0058
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)	5d. PROJECT NUMBER
Mark Costello and Jonathan Rogers	1L1612618AH
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
U.S. Army Research Laboratory ATTN: RDRL-WML-E Aberdeen Proving Ground, MD 21005	ARL-CR-670

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report documents the theory and use of the BOOM computer program. BOOM is an exterior ballistics simulation program specifically designed to predict atmospheric flight of smart projectiles. The software contains a projectile dynamic model coupled to an open structure flight control system. The code can be run on PC, Unix, or Mac systems.

15. SUBJECT TERMS

projectiles, trajectory, aeroballistics, flight mechanics, smart projectiles

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Mark Costello
Unclassified	Unclassified	Unclassified	UU	78	19b. TELEPHONE NUMBER (*Include area code*) (410) 306-0800

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

List of Tables

INTENTIONALLY LEFT BLANK.

1. Introduction

BOOM is a FORTRAN computer program that was specifically designed to predict the atmospheric flight mechanics of smart projectile systems. The key elements of the package are a dynamic model of the projectile in atmospheric flight and a dynamic model of the projectile control system. The software automatically couples the projectile and its associated control system and subsequently allows simulation of the time response of the coupled system as well as computation of impact dispersion statistics. A cartoon of the analysis flow in BOOM is shown in figure 1.

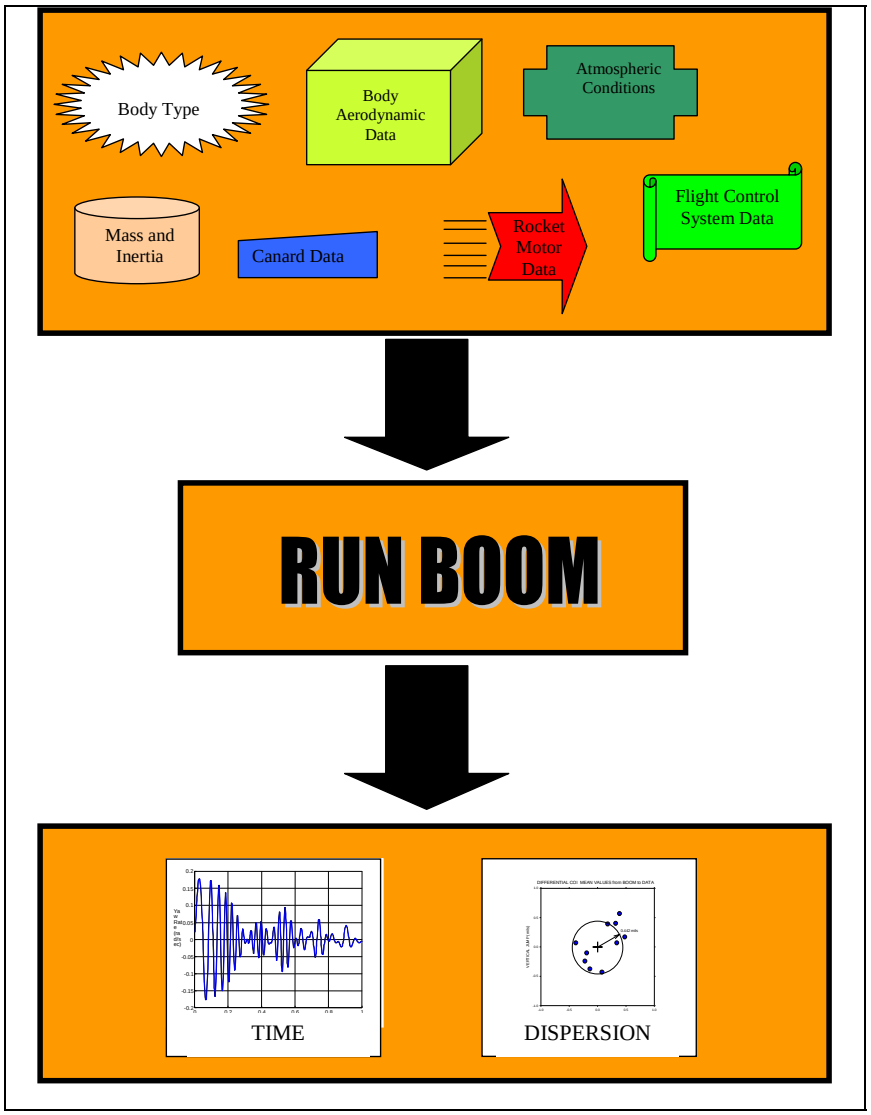

Figure 1. BOOM analysis schematic.

BOOM utilizes a 6-degrees-of-freedom (DOF) rigid projectile model to represent the inertial position and orientation of the projectile body. The 3 translation DOF are the inertial position components of the projectile mass center. Euler angles are used to represent the rotation DOF. The dynamic model allows for a completely populated inertia matrix, thus allowing modeling of mass unbalanced projectile configurations. Also, the projectile body model includes a sophisticated body aerodynamic model consisting of steady and unsteady aerodynamic terms. The Projectile Rocket Ordnance Design and Analysis System (PRODAS) aerodynamic expansion is utilized for body aerodynamics. The projectile body can also incorporate an arbitrary number of rocket motors and canard lifting surfaces. The rocket motors and canards can be placed at any location on the body. Changes in projectile body mass center and inertia matrix are properly accounted for as the rocket motors burn.

The projectile control system uses an open-structure architecture to describe the control system connectivity. A wide variety of control system building blocks is available, including gain, sum, multiply, state space filters, polynomial filters, trigonometric functions, triggers, sample and hold, accelerometers, inertial to body transformations, single-axis transformations, constants, table look ups, magnitude and phase, etc. The user constructs a control system by appropriately arranging control system building blocks. The software automatically couples all control system elements together. Any physical parameter of the projectile model can be dynamically controlled. With this arrangement, virtually any projectile flight control system can be modeled in detail.

This report outlines the theory and methodology behind the use of BOOM. Details on the projectile dynamic model and the flight control system model are provided. The procedure for running BOOM is also outlined, with input data files described in the appendices. Example trajectories and control system files will be provided in a future report.

2. Projectile Flight Dynamic Model

The mathematical model describing projectile motion relies on rigid body dynamics. The model assumes that the surface of the Earth is an inertial reference frame. It permits body aerodynamic, canard, rocket thrust, and weight forces and moments to be applied to the projectile. Section 2.1 describes the underlying equations of motion for a rigid body projectile. Section 2.2 describes the body aerodynamic model, while section 2.3 details the canard model. Section 2.4 derives the rocket motor model, and section 2.5 describes injection forces and moments.

2.1 Equations of Motion

The dynamic equations presented in this section use the ground surface as an inertial reference frame. Also, the mass center can be arbitrarily placed on the body and the inertia matrix is completely general, allowing for off-diagonal inertia matrix elements. Rigid body projectile

motion consists of 3 translation DOF and 3 rotation DOF. The 3 translation DOF are the inertial position components of the projectile mass center. Euler angles are used to parameterize rigid body rotation. The standard sequence of rotations used in air vehicle flight mechanics is employed, i.e., a body-fixed set of rotations that begins from the inertial axis and subsequently executes a yaw, then pitch, and then roll rotation. The body frame is defined in the conventional manner, and the dynamic equations are expressed in this coordinate system. The translation kinematics, rotation kinematics, translation dynamics, and rotation dynamics for the rigid projectile are given by equations 1–4. The following equations are well known and reported in many sources.[1]

$$\begin{Bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{Bmatrix} = \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix} \begin{Bmatrix} u \\ v \\ w \end{Bmatrix}, \tag{1}$$

$$\begin{Bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Bmatrix} = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi / c_\theta & c_\phi / c_\theta \end{bmatrix} \begin{Bmatrix} p \\ q \\ r \end{Bmatrix}, \tag{2}$$

$$\begin{Bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{Bmatrix} = \begin{Bmatrix} X/m \\ Y/m \\ Z/m \end{Bmatrix} - \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \begin{Bmatrix} u \\ v \\ w \end{Bmatrix}, \tag{3}$$

and

$$\begin{Bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{Bmatrix} = \begin{bmatrix} I_{XX} & I_{XY} & I_{XZ} \\ I_{XY} & I_{YY} & I_{YZ} \\ I_{XZ} & I_{YZ} & I_{ZZ} \end{bmatrix}^{-1} \left[\begin{Bmatrix} L \\ M \\ N \end{Bmatrix} - \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \begin{bmatrix} I_{XX} & I_{XY} & I_{XZ} \\ I_{XY} & I_{YY} & I_{YZ} \\ I_{XZ} & I_{YZ} & I_{ZZ} \end{bmatrix} \begin{Bmatrix} p \\ q \\ r \end{Bmatrix} \right]. \tag{4}$$

In equation 4, the inertia matrix is with respect to the projectile body axis at the mass center, as shown in equation 5 as follows:

$$[I] = \begin{bmatrix} I_{XX} & I_{XY} & I_{XZ} \\ I_{XY} & I_{YY} & I_{YZ} \\ I_{XZ} & I_{YZ} & I_{ZZ} \end{bmatrix}. \tag{5}$$

[1] Etkin, B. *Dynamics of Atmospheric Flight*; John Wiley and Sons: New York, NY, 1972.

As shown in equations 6 and 7, the total applied force and moment are split into contributions due to weight ($_W$), body aerodynamic force ($_A$), canard loads ($_C$), rocket motor thrust ($_R$), and injection force ($_I$), respectively, as follows:

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix} = \begin{Bmatrix} X_W \\ Y_W \\ Z_W \end{Bmatrix} + \begin{Bmatrix} X_A \\ Y_A \\ Z_A \end{Bmatrix} + \begin{Bmatrix} X_C \\ Y_C \\ Z_C \end{Bmatrix} + \begin{Bmatrix} X_R \\ Y_R \\ Z_R \end{Bmatrix} + \begin{Bmatrix} X_I \\ Y_I \\ Z_I \end{Bmatrix}, \tag{6}$$

and

$$\begin{Bmatrix} L \\ M \\ N \end{Bmatrix} = \begin{Bmatrix} L_A \\ M_A \\ N_A \end{Bmatrix} + \begin{Bmatrix} L_C \\ M_C \\ N_C \end{Bmatrix} + \begin{Bmatrix} L_R \\ M_R \\ N_R \end{Bmatrix} + \begin{Bmatrix} L_I \\ M_I \\ N_I \end{Bmatrix}. \tag{7}$$

The body aerodynamic, canard, and rocket thrust models are detailed in sections 2.2–2.4. Injection forces and moments are described in section 2.5, and weight force is given in equation 8 as follows:

$$\begin{Bmatrix} X_W \\ Y_W \\ Z_W \end{Bmatrix} = W \begin{Bmatrix} -s_\theta \\ s_\phi c_\theta \\ c_\phi c_\theta \end{Bmatrix}. \tag{8}$$

The mathematical model and implementation in BOOM described by equations 1–8 has been validated against spark range data for a generic 25-mm fin-stabilized, sabot-launched projectile.[2] Agreement between the model and range data is excellent.

2.2 Body Aerodynamic Model

The aerodynamic forces and moment model is based on the PRODAS aerodynamic expansion. This model is employed extensively in the exterior ballistics community and represents a well-accepted standard. The PRODAS aerodynamic force and moment expansion is valid for both fin- and spin-stabilized symmetric and slightly non-symmetric projectiles operating at relatively small angles of attack. As shown in equation 9, the aerodynamic forces on the projectile are split into standard steady ($_{SA}$) and Magnus ($_{MA}$) terms as follows:

[2] Costello, M. F.; Anderson, D. A. Effect of Internal Unbalance on the Stability and Terminal Accuracy of a Field Artillery Projectile. *Proceedings of the 1996 AIAA Atmospheric Flight Mechanics Conference*, San Diego, CA, 1996.

$$\begin{Bmatrix} X_A \\ Y_A \\ Z_A \end{Bmatrix} = \begin{Bmatrix} X_{SA} \\ Y_{SA} \\ Z_{SA} \end{Bmatrix} + \begin{Bmatrix} X_{MA} \\ Y_{MA} \\ Z_{MA} \end{Bmatrix}. \tag{9}$$

Equations 10–15 show the expansion of each component of the forces and moments in terms of PRODAS aerodynamic coefficients as follows:

$$X_{SA} = -\frac{\pi}{8}\rho V_A^2 D^2 (C_{X0} + C_{X2}\varepsilon^2), \tag{10}$$

$$Y_{SA} = -\frac{\pi}{8}\rho V_A^2 D^2 \left[C_{Y0} + \bar{C}_{NA}\frac{v_A}{V_A} + \bar{C}_{NG}\cos(N_F\varphi_\alpha)\frac{v_A}{V_A} - \bar{C}_{NG}\sin(N_F\varphi_\alpha)\frac{w_A}{V_A} \right], \tag{11}$$

$$Z_{SA} = -\frac{\pi}{8}\rho V_A^2 D^2 \left[C_{Z0} + \bar{C}_{NA}\frac{w_A}{V_A} + \bar{C}_{NG}\cos(N_F\varphi_\alpha)\frac{w_A}{V_A} + \bar{C}_{NG}\sin(N_F\varphi_\alpha)\frac{v_A}{V_A} \right], \tag{12}$$

$$X_{MA} = 0, \tag{13}$$

$$Y_{MA} = \frac{\pi}{8}\rho V_A^2 D^2 \frac{pD}{2V_A}\left[C_{YPA}\frac{w_A}{V_A} \right], \tag{14}$$

and

$$Z_{MA} = -\frac{\pi}{8}\rho V_A^2 D^2 \frac{pD}{2V_A}\left[C_{YPA}\frac{v_A}{V_A} \right]. \tag{15}$$

In equations 11 and 12, $\bar{C}_{NG}$ is the roll-induced side force caused by body fins, N_F is the number of fins, and φ_α is the fin angular spacing about the roll axis. As shown in equation 16, the total applied body moments contain steady ($_{SA}$), unsteady ($_{UA}$), and Magnus ($_{MA}$) terms as follows:

$$\begin{Bmatrix} L \\ M \\ N \end{Bmatrix} = \begin{Bmatrix} L_{SA} \\ M_{SA} \\ N_{SA} \end{Bmatrix} + \begin{Bmatrix} L_{UA} \\ M_{UA} \\ N_{UA} \end{Bmatrix} + \begin{Bmatrix} L_{MA} \\ M_{MA} \\ N_{MA} \end{Bmatrix}. \tag{16}$$

Equation 17 shows that the steady body aerodynamic moment is computed with a cross product between the distance vector from the center of gravity to the center of pressure and the steady body aerodynamic force vector (shown previously) as follows:

$$\begin{Bmatrix} \boldsymbol{L}_{SA} \\ \boldsymbol{M}_{SA} \\ \boldsymbol{N}_{SA} \end{Bmatrix} = \begin{bmatrix} 0 & \boldsymbol{WL}_{CG} - \boldsymbol{WL}_{COP} & \boldsymbol{BL}_{COP} - \boldsymbol{BL}_{CG} \\ \boldsymbol{WL}_{COP} - \boldsymbol{WL}_{CG} & 0 & \boldsymbol{SL}_{CG} - \boldsymbol{SL}_{COP} \\ \boldsymbol{BL}_{CG} - \boldsymbol{BL}_{COP} & \boldsymbol{SL}_{COP} - \boldsymbol{SL}_{CG} & 0 \end{bmatrix} \begin{Bmatrix} \boldsymbol{X}_{SA} \\ \boldsymbol{Y}_{SA} \\ \boldsymbol{Z}_{SA} \end{Bmatrix}. \tag{17}$$

Likewise, the Magnus aerodynamic moment is computed with a cross product between the distance vector from the center of mass to the center of Magnus force and the Magnus force vector given by equation 18 as follows:

$$\begin{Bmatrix} \boldsymbol{L}_{MA} \\ \boldsymbol{M}_{MA} \\ \boldsymbol{N}_{MA} \end{Bmatrix} = \begin{bmatrix} 0 & \boldsymbol{WL}_{CG} - \boldsymbol{W\bar{L}}_{MAG} & \boldsymbol{B\bar{L}}_{MAG} - \boldsymbol{BL}_{CG} \\ \boldsymbol{W\bar{L}}_{MAG} - \boldsymbol{WL}_{CG} & 0 & \boldsymbol{SL}_{CG} - \boldsymbol{S\bar{L}}_{MAG} \\ \boldsymbol{BL}_{CG} - \boldsymbol{B\bar{L}}_{MAG} & \boldsymbol{S\bar{L}}_{MAG} - \boldsymbol{SL}_{CG} & 0 \end{bmatrix} \begin{Bmatrix} \boldsymbol{X}_{MA} \\ \boldsymbol{Y}_{MA} \\ \boldsymbol{Z}_{MA} \end{Bmatrix}. \tag{18}$$

In the previous equations, the stationline (*SL*), buttline (*BL*), and waterline (*WL*) displacement components are measured from coordinate systems with an origin at the base of the projectile that is aligned to the body axes. The following subscript conventions should be noted: $_{CG}$ denotes center of gravity, $_{COP}$ denotes center of pressure, and $_{MAG}$ denotes center of Magnus force. The unsteady body aerodynamic moment provides a damping source for projectile angular motion and is given by equations 19–21 as follows:

$$\boldsymbol{L}_{UA} = \frac{\pi}{8} \rho \boldsymbol{V}_A^2 \boldsymbol{D}^3 \left[\boldsymbol{C}_{LDD} + \frac{\boldsymbol{pD}}{2\boldsymbol{V}_A} \boldsymbol{C}_{LP} \right], \tag{19}$$

$$\boldsymbol{M}_{UA} = \frac{\pi}{8} \rho \boldsymbol{V}_A^2 \boldsymbol{D}^3 \left[\frac{\boldsymbol{qD}}{2\boldsymbol{V}_A} \bar{\boldsymbol{C}}_{MQ} \right], \tag{20}$$

and

$$\boldsymbol{N}_{UA} = \frac{\pi}{8} \rho \boldsymbol{V}_A^2 \boldsymbol{D}^3 \left[\frac{\boldsymbol{rD}}{2\boldsymbol{V}_A} \bar{\boldsymbol{C}}_{MQ} \right]. \tag{21}$$

Equations 10–21 utilize the following intermediate expressions:

$$\boldsymbol{S\bar{L}}_{MAG} = \boldsymbol{SL}_{MAG_0} + \boldsymbol{SL}_{MAG_2} \varepsilon^2 + \boldsymbol{SL}_{MAG_4} \varepsilon^4, \tag{22}$$

$$\bar{BL}_{MAG} = BL_{MAG_0} + BL_{MAG_2}\varepsilon^2 + BL_{MAG_4}\varepsilon^4, \tag{23}$$

$$\bar{WL}_{MAG} = WL_{MAG_0} + WL_{MAG_2}\varepsilon^2 + WL_{MAG_4}\varepsilon^4, \tag{24}$$

$$\bar{C}_{NG} = C_{NG_3}\varepsilon^2, \tag{25}$$

$$\bar{C}_{NA} = C_{NA_1} + C_{NA_3}\varepsilon^2 + C_{NG_3}\varepsilon^2 \cos(N_F \varphi_\alpha), \tag{26}$$

$$\bar{C}_{MQ} = C_{MQ} + C_{MQ_2}\varepsilon^2, \tag{27}$$

and

$$\varepsilon = \frac{\sqrt{v_A^2 + w_A^2}}{\sqrt{u_A^2 + v_A^2 + w_A^2}}. \tag{28}$$

The following aerodynamic coefficients and aerodynamic center distances are all a function of the local Mach number at the center of mass of the projectile: C_{X0}, C_{X2}, C_{Y0}, C_{Z0}, C_{NA_1}, C_{NA_3}, C_{NG_3}, C_{YPA}, C^L_{ZA3}, C_{MQ}, C_{MQ_2}, C_{LDD}, C_{LP}, SL_{COP}, BL_{COP}, WL_{COP}, SL_{MAG_0}, BL_{MAG_0}, WL_{MAG_0}, SL_{MAG_2}, BL_{MAG_2}, WL_{MAG_2}, SL_{MAG_4}, BL_{MAG_4}, and WL_{MAG_4}. Computationally, these Mach number dependent parameters are obtained by a table lookup scheme using linear interpolation. Mach number is computed at the center of gravity of the respective projectile. The coefficient values are obtained for specific projectile shapes using existing empirical missile and projectile aerodynamic databases.

2.3 Canard Model

In BOOM, an arbitrary number of canards can be placed at different locations on any projectile body to model aerodynamic lifting surfaces. The aerodynamic force due to a single canard is modeled as a point force acting at the lifting surface aerodynamic center. In specifying the location of the lifting surface, the user specifies the application point of the canard force. The orientation of a particular canard is obtained by a set of three body-fixed rotations. Starting with the canard axis aligned with the projectile body axis, the canard is rotated about the $\vec{i}_B$ axis by the azimuthal angle (ϕ_{C_i}) and then about the resulting intermediate $\vec{k}$ axis by the sweep angle (γ_{C_i}). If all angles are zero, the lifting surface is in the $\vec{i}_B - \vec{j}_B$ plane. The transformation from the *ith* canard reference frame to the parent projectile body axis is given by equation 29 as follows:

$$\boldsymbol{T}_{C_i} = \begin{bmatrix} \cos(\gamma_{C_i}) & -\sin(\gamma_{C_i}) & 0 \\ \cos(\phi_{C_i})\sin(\gamma_{C_i}) & \cos(\phi_{C_i})\cos(\gamma_{C_i}) & -\sin(\phi_{C_i}) \\ \sin(\phi_{C_i})\sin(\gamma_{C_i}) & \sin(\phi_{C_i})\cos(\gamma_{C_i}) & \cos(\phi_{C_i}) \end{bmatrix}. \tag{29}$$

Because T_{C_i} is an orthogonal matrix, the inverse transformation from the parent projectile body axis to the *ith* canard reference frame is simply the matrix transpose of T_{C_i}. Strip theory is used to compute the canard aerodynamic loads. Figure 2 provides a diagram of the canard aerodynamic force field.

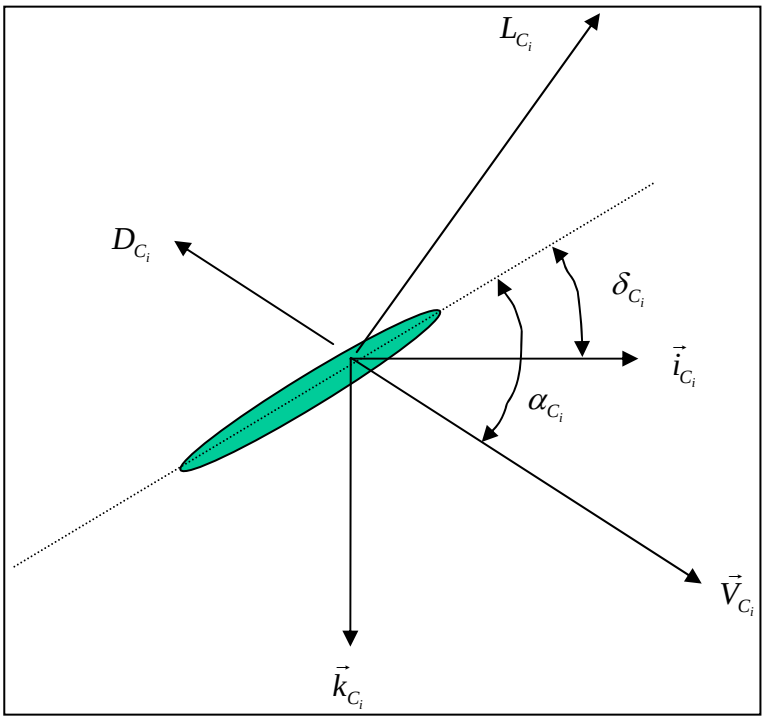

Figure 2. Canard aerodynamic model force diagram.

Notice that the aerodynamic angle of attack of the *ith* canard is calculated using only the $u_{A_{C_i}}$ and $w_{A_{C_i}}$ components of the relative air velocity experienced by the canard computation point. In the parent projectile body axis, the *ith* canard force is given by equation 30 as follows:

$$\begin{Bmatrix} \boldsymbol{X}_{C_i} \\ \boldsymbol{Y}_{C_i} \\ \boldsymbol{Z}_{C_i} \end{Bmatrix} = \boldsymbol{q}_{C_i} \boldsymbol{S}_{C_i} \left[\boldsymbol{T}_{C_i} \right] \begin{Bmatrix} \boldsymbol{C}_{L_{C_i}} \sin(\alpha_{C_i} - \delta_{C_i}) - \boldsymbol{C}_{D_{C_i}} \cos(\alpha_{C_i} - \delta_{C_i}) \\ 0 \\ -\boldsymbol{C}_{L_{C_i}} \cos(\alpha_{C_i} - \delta_{C_i}) - \boldsymbol{C}_{D_{C_i}} \sin(\alpha_{C_i} - \delta_{C_i}) \end{Bmatrix}. \tag{30}$$

In equation 30, q_{C_i} is the dynamic pressure at the canard computation point, S_i is the canard reference area, and δ_{C_i} is the canard pitch angle. Typical smart munitions that employ canards actively control the canard pitch angle. Dynamic pressure is given by equation 31 as follows:

$$\boldsymbol{q}_{C_i} = \frac{1}{2}\rho\left(\boldsymbol{u}^2_{A_{C_i}} + \boldsymbol{v}^2_{A_{C_i}} + \boldsymbol{w}^2_{A_{C_i}}\right). \tag{31}$$

The canard lift and drag coefficients are expanded in terms of canard aerodynamic angle of attack and local Mach number at the canard computation point in equations 32 and 33 as follows:

$$\boldsymbol{C}_{L_{C_i}} = \boldsymbol{C}_{L1_{C_i}}\alpha_{C_i} + \boldsymbol{C}_{L3_{C_i}}\alpha^3_{C_i} + \boldsymbol{C}_{L5_{C_i}}\alpha^5_{C_i}, \tag{32}$$

and

$$\boldsymbol{C}_{D_{C_i}} = \boldsymbol{C}_{D0_{C_i}} + \boldsymbol{C}_{D2_{C_i}}\alpha^2_{C_i} + \boldsymbol{C}_{I_{C_i}}\boldsymbol{C}^2_{L_{C_i}}. \tag{33}$$

The coefficients in equations 32 and 33 are Mach number dependent. Canard angle of attack is computed using equation 34 using the local relative velocity at the canard computation point as follows:

$$\alpha_{C_i} = \mathbf{tan}^{-1}\left(\frac{\boldsymbol{w}_{A_{C_i}}}{\boldsymbol{u}_{A_{C_i}}}\right). \tag{34}$$

2.4 Rocket Motor Model

In BOOM, an arbitrary number of rocket motors can be placed at different locations on any projectile body to model the thrust force generated by a burning rocket motor. The force due to a single rocket motor is modeled as a point force acting at the rocket motor computation point. The rocket motor application point is specified using input data. In the parent projectile body axis, the *ith* rocket motor force is given by equation 35 as follows:

$$\begin{Bmatrix} X_{R_i} \\ Y_{R_i} \\ Z_{R_i} \end{Bmatrix} = A_{T_i} T_{R_i} \begin{Bmatrix} N_{RX_i} \\ N_{RY_i} \\ N_{RZ_i} \end{Bmatrix}. \tag{35}$$

In equation 35, the rocket motor thrust, T_{R_i}, is time dependent and given by a table of data. Thrust at a particular instant is computed by linear interpolation of the tabular data. For time values outside the tabular data, the nearest thrust value is used.

In a standard rocket assisted projectile, the base drag of the projectile is reduced when the rocket is burning. In BOOM, the aerodynamic model requires two tables of C_{X0}. One table is used when the motor is off, and the other model is used when the rocket is on.

2.5 Atmosphere Model

The atmosphere model in BOOM specifies the air density and speed of sound of the atmosphere as a function of altitude, as well as the mean atmospheric wind. The following three options are available to specify density and speed of sound: constant density and speed of sound, equation form for density and speed of sound using the standard atmosphere equations, or a table form for density and speed of sound. The constant density and speed of sound option is useful when correlating code results with range data. The equation density and speed of sound model is favored in general parametric trade studies. The table form is useful in evaluating the effect of various atmospheric conditions on projectile flight.

Equations 36 and 37 provide the equations for air density and speed of sound used in the equation model as follows:

$$\rho = \begin{Bmatrix} 0.0023784722\left[1 - 0.0000068789\boldsymbol{z}\right]^{4.258} & \boldsymbol{z} > -35,332\ \boldsymbol{ft} \\ 0.00072674385\mathrm{e}^{0.0000478(z+35,332)} & \boldsymbol{z} \le -35,332\ \boldsymbol{ft} \end{Bmatrix}, \tag{36}$$

and

$$\boldsymbol{a} = \begin{Bmatrix} 49.0124\sqrt{518.4 + 0.003566\boldsymbol{z}} & \boldsymbol{z} > -35,332\ \boldsymbol{ft} \\ 970.8985166 & \boldsymbol{z} \le -35,332\ \boldsymbol{ft} \end{Bmatrix}. \tag{37}$$

A simple atmospheric mean wind model is available in BOOM. The mean wind is assumed to be horizontal, i.e., in the plane formed by $\vec{i}_I$ and $\vec{j}_I$. As shown in figure 3, the mean wind is directed at an angle ψ_W from the inertial reference frame.

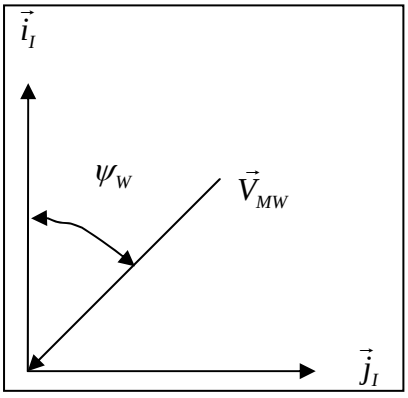

Figure 3. Mean atmospheric wind velocity diagram.

The mean wind vector is given by equation 38 as follows:

$$\vec{\boldsymbol{V}}_{\boldsymbol{MW}} = \boldsymbol{V}_{\boldsymbol{MW}} \cos(\psi_W)\vec{\boldsymbol{i}}_{\boldsymbol{I}} + \boldsymbol{V}_{\boldsymbol{MW}} \sin(\psi_W)\vec{\boldsymbol{j}}_{\boldsymbol{I}}. \tag{38}$$

The magnitude of the mean wind velocity is a function of altitude to model the Earth's boundary layer. Equation 39 provides this relationship as follows:

$$\boldsymbol{V_{MW}} = 0.636619\sigma_{\boldsymbol{MW}}\ \mathbf{tan}^{-1}\left(\frac{\boldsymbol{z}}{1000}\right). \tag{39}$$

If the projectile is fired with $\psi_0 = 0$ and $\psi_W = 0$, the projectile experiences a headwind, while if $\psi_0 = 0$ and $\psi_W = 90^\circ$, the projectile experiences a left crosswind.

2.6 Injection Forces and Moments

Injection forces and moments are included in BOOM in order to simulate control forces and moments without having to specify a means to exert control force. This technique can be useful when sizing control mechanisms to determine control force magnitudes needed to achieve a certain control authority. Furthermore, injection forces and moments are helpful during preliminary flight control design when a detailed canard or rocket model may not be available, but control authority estimates are still desired. The user is responsible for defining injection forces and moments within the flight control system, which are then included in the dynamic simulation by setting global injection force variables (XINJECTFORCE, YINJECTFORCE, and ZINJECTFORCE) and injection moment variables (LINJECTMOMENT, MINJECTMOMENT, and NINJECTMOMENT) to desired values within the flight controller.

3. Flight Control System Model

BOOM has been designed to simulate smart projectile systems. Control of a projectile can be achieved in many different ways. For example, an extended-range projectile might be guided with canards. To control the projectile, the canard pitch angle is changed in flight depending on the location and orientation of the projectile. In a different application, the projectile might be controlled by pulse jets that are activated during flight. Because many different projectile flight control mechanisms are possible, a general smart munition simulation tool must be able to modify many model parameters during flight. This is achieved in BOOM by allowing the user to dynamically control any parameter that is stored in global memory. Not only are many different control mechanisms used to control projectiles, many different strategies are employed to guide and control a projectile. The control system strategy is conveyed through the control law. The control law stipulates a set of operations performed on sensor data to determine how the controls should be changed in flight. The language to describe a control law is the block diagram. While many different control laws can be created, all can be expressed in terms of a block diagram. For this reason, BOOM uses an open structure flight control system modeling architecture that enables flight control system models to be built directly from block diagram information. The

open structure flight control system consists of a set of basic building blocks called flight control system modeling elements. Through program input data, the user selects and arranges flight control system modeling elements to mimic the physical arrangement that is desired to simulate. The user matches appropriate flight control system element outputs with physical control parameters. At any given time instant, the controlled parameters are computed as shown in equation 40 as follows:

$$\vec{\boldsymbol{P}}_{\boldsymbol{C}} = \vec{\boldsymbol{y}}_{\boldsymbol{FCS}} \,. \tag{40}$$

In equation 40, the vector of physical control parameters is denoted as $\vec{P}_C$, while the vector of controlled parameters is denoted as $\vec{\boldsymbol{y}}_{\boldsymbol{FCS}}$. The following section describes the various flight control system modeling elements that are available in BOOM, while section 3.2 describes an example flight control system.

3.1 Modeling Elements

There are currently 52 flight control system modeling elements for a user to choose from to construct a flight control system.

3.1.1 Gain Element

The gain element simply multiplies an input signal by a constant to generate the output signal. This element is represented mathematically by equation 41 as follows:

$$\boldsymbol{y}_{\boldsymbol{FCS}} = \boldsymbol{K}\boldsymbol{u}_{\boldsymbol{FCS}} \,. \tag{41}$$

The gain element is a single-input-single-output modeling element. The value for the gain, K, is specified by the user.

3.1.2 Sum Element

The sum element adds together a set of input signals that are first multiplied by a gain. This element is represented mathematically by equation 42 as follows:

$$\boldsymbol{y}_{\boldsymbol{FCS}} = \boldsymbol{K}_1\boldsymbol{u}_{\boldsymbol{FCS}_1} + \boldsymbol{K}_2\boldsymbol{u}_{\boldsymbol{FCS}_2} + \boldsymbol{...} + + \boldsymbol{K}_N\boldsymbol{u}_{\boldsymbol{FCS}_N} \,. \tag{42}$$

The sum element is a multiple-input-single-output modeling element. The value for the gains, K_i, is specified by the user.

3.1.3 Multiply Element

The multiply element multiplies a set of input signals. The input signals are not pre-multiplied by a gain. This element is represented mathematically by equation 43 as follows:

$$\boldsymbol{y}_{\boldsymbol{FCS}} = \boldsymbol{u}_{\boldsymbol{FCS}_1}\boldsymbol{u}_{\boldsymbol{FCS}_2}\boldsymbol{...}\boldsymbol{u}_{\boldsymbol{FCS}_N} \,. \tag{43}$$

The multiply element is a multiple-input-single-output modeling element.

3.1.4 Square Root Element

The square root element takes the square root of the input signal. This element is represented mathematically by equation 44 as follows:

$$\boldsymbol{y}_{FCS} = \sqrt{\boldsymbol{u}_{FCS}} \ . \tag{44}$$

The square root element is a single-input-single-output modeling element.

3.1.5 Cube Element

The cube element outputs the cube of the input signal. This element is represented mathematically by equation 45 as follows:

$$\boldsymbol{y}_{FCS} = \boldsymbol{u}_{FCS}^{3} \, . \tag{45}$$

The cube element is a single-input-single-output modeling element.

3.1.6 Absolute Value Element

The absolute value element takes the absolute value of the input signal. This element is represented mathematically by equation 46 as follows:

$$\boldsymbol{y}_{FCS} = \left| \boldsymbol{u}_{FCS} \right| . \tag{46}$$

The absolute value element is a single-input-single-output modeling element.

3.1.7 Sine Element

The sine element computes the trigonometric sine of the input signal. This function assumes the input signal is in radians. This element is represented mathematically by equation 47 as follows:

$$\boldsymbol{y}_{FCS} = \mathbf{sin}\left(\boldsymbol{u}_{FCS} \right) . \tag{47}$$

The sine element is a single-input-single-output modeling element.

3.1.8 Cosine Element

The cosine element computes the trigonometric cosine of the input signal. This function assumes the input signal is in radians. This element is represented mathematically by equation 48 as follows:

$$\boldsymbol{y}_{FCS} = \mathbf{cos}\left(\boldsymbol{u}_{FCS} \right) . \tag{48}$$

The cosine element is a single-input-single-output modeling element.

3.1.9 Tangent Element

The tangent element computes the trigonometric tangent of the input signal. This function assumes the input signal is in radians. This element is represented mathematically by equation 49 as follows:

$$\boldsymbol{y_{FCS}} = \mathbf{tan}\left(\boldsymbol{u_{FCS}}\right). \tag{49}$$

The tangent element is a single-input-single-output modeling element.

3.1.10 Arc Sine Element

The arc sine element computes the trigonometric arc sine of the input signal. The output is in radians. This element is represented mathematically by equation 50 as follows:

$$\boldsymbol{y_{FCS}} = \mathbf{sin}^{-1}\left(\boldsymbol{u_{FCS}}\right). \tag{50}$$

The arc sine element is a single-input-single-output modeling element.

3.1.11 Arc Cosine Element

The arc cosine element computes the trigonometric arc cosine of the input signal. The output is in radians. This element is represented mathematically by equation 51 as follows:

$$\boldsymbol{y_{FCS}} = \mathbf{cos}^{-1}\left(\boldsymbol{u_{FCS}}\right). \tag{51}$$

The arc cosine element is a single-input-single-output modeling element.

3.1.12 Arc Tangent Element

The arc tangent element computes the trigonometric arc tangent of the input signal. The output is in radians. This element is represented mathematically by equation 52 as follows:

$$\boldsymbol{y_{FCS}} = \mathbf{tan}^{-1}\left(\boldsymbol{u_{FCS}}\right). \tag{52}$$

The arc tangent element is a single-input-single-output modeling element.

3.1.13 Quantization Element

Given an input signal, the quantization element represents the number as a digital computer would with a finite word length. Thus, the quantizer chops the input signal like an analog to digital converter. In base 2, the input signal can be represented as shown in equation 53 as follows:

$$\boldsymbol{u_{FCS}} = \pm\left(\alpha_{-N}\,2^{-N} + \alpha_{-N+1}\,2^{-N+1} + \ldots + \alpha_0 + \ldots + \alpha_{N-1}\,2^{N-1} + \alpha_N\,2^{N}\right). \tag{53}$$

In a fixed point representation, only certain powers of two are allowed to be retained in the output. Each power of two that is retained in the expansion requires 1 byte of storage. Both the number of bytes before and after the decimal place must be specified. Provided the input signal

is in the range of the fixed point representation, equation 54 provides the mathematical formula for fixed point quantization as follows:

$$y_{FCS} = \frac{u_{FCS}}{|u_{FCS}|}\,\mathbf{int}(|u_{FCS}|\,2^{E_B})/2^{E_B}. \tag{54}$$

In equation 54, E_B is the number of bytes retained after the decimal. If the input signal is out of range of the fixed point representation, then the nearest number that can be represented with the fixed point model is output. Equations 55–59 provide the mathematical formulas for floating point quantization as follows:

$$y_{FCS} = A_{FP}\,2^{E_{FP}}, \tag{55}$$

$$A_{FP} = \mathbf{int}\left(2^{\mathbf{mod}(\log(|u_{FCS}|)/\log(2),1)}2^{A_B}\right)/2^{A_B}, \tag{56}$$

$$B_{FP} = \mathbf{int}\left(\log(|u_{FCS}|)/\log(2)\right), \tag{57}$$

$$C_{FP} = \mathbf{int}(\log(|B_{FP}|)/\log(2)) + 1, \tag{58}$$

and

$$E_{FP} = 2^{C_{FP}-E_B}\,\mathbf{int}\left(B_{FP}/2^{C_{FP}-E_B}\right). \tag{59}$$

In equations 55–59, A_{FP} is the quantized representation of the mantissa of the floating point number, and A_B is the number of bytes retained for the mantissa. Also, E_{FP} is the quantized representation of the exponent, and E_B is the number of bytes retained in the exponent. The quantization element is a single-input-single-output modeling element.

3.1.14 Magnitude and Phase Element

The magnitude and phase element converts a set of Cartesian coordinates to a polar representation. The two input signals are multiplied by a gain before the magnitude and phase angle is calculated. The phase angle is computed in radians. This element is represented mathematically by equations 60 and 61 as follows:

$$y_{FCS_1} = \tan^{-1}\left(K_1 u_{FCS_1}, K_2 u_{FCS_2}\right), \tag{60}$$

and

$$y_{FCS_2} = \sqrt{\left(K_1 u_{FCS_1}\right)^2 + \left(K_2 u_{FCS_2}\right)^2}. \tag{61}$$

It should be noted in equation 60 that the arc tangent will be resolved into the proper quadrant. The magnitude and phase element is a two-input-two-output modeling element.

3.1.15 RMS Element

The RMS element outputs the root mean squared of a signal with three inputs. This element is represented mathematically by equation 62 as follows:

$$y_{FCS_1} = \sqrt{u_{FCS_1}^2 + u_{FCS_2}^2 + u_{FCS_3}^2} \; . \tag{62}$$

The RMS element is a three-input-single-output modeling element.

3.1.16 Inversion Element

The inversion element computes the inverse of the input signal. This element is represented mathematically by equation 63 as follows:

$$y_{FCS} = \frac{1}{u_{FCS}} . \tag{63}$$

Care must be taken when using the inversion element since the element will generate an infinite output if the input signal is zero. The inversion element is a single-input-single-output modeling element.

3.1.17 Exponential Element

The exponential element outputs the exponential of the input signal. This element is represented mathematically by equation 64 as follows:

$$y_{FCS} = e^{u_{FCS}} \; . \tag{64}$$

The exponential element is a single-input-single-output modeling element.

3.1.18 Constant Element

The constant element generates a constant output. The value of the constant is input by the user. This element is represented mathematically by equation 65 as follows:

$$y_{FCS} = C \; . \tag{65}$$

The constant element is a zero-input-single-output modeling element.

3.1.19 Modulo Element

The modulo element outputs a value equal to the modulo of the input value, with a given parameter value. The modulo of the input value is defined as the remainder after the input is divided by the parameter value. For instance, with an input of 7 and a parameter value of 3, the output of the modulo element would be 1. The modulo element is a single-input-single-output modeling element.

3.1.20 Trigger Element

The trigger element acts like a switch and generates an output signal that is either 0 or 1. When a simulation begins, the trigger element is set to 0. Once the input signal exceeds a prescribed value in absolute value, the trigger is tripped and the element output becomes 1. After the trigger is tripped, the element output will remain equal to 1 even if the input signal falls below the prescribed trigger level. The trigger element is a single-input-single-output modeling element.

3.1.21 Switch Element

The switch element outputs a signal that is either 0 or 1. The element outputs 0 until the trigger level is reached, and then it will have an output of 1. The level falls back to 0 if the input level drops below the threshold. The switch element is a single-input-single-output modeling element.

3.1.22 Positive Trigger Element

The positive trigger element outputs a signal that is either 0 or 1. The positive trigger has an output of 0 until the trigger level is reached on a rising signal, and then it has an output of 1. Once triggered, this control cannot be reset. It will not be triggered by a constant and/or falling signal, even if it is above the threshold. The positive trigger element is a single-input-single-output modeling element.

3.1.23 Hold Element

The hold element acts like a switch and changes the nature of the output signal when a trigger has been tripped. The hold element has two input signals. The first input signal is used to determine when the trigger has been tripped. The element initially generates an output signal that is equal to input signal number 2. Once input signal number 1 exceeds a prescribed value in absolute value, a trigger is tripped and the element output becomes fixed at the value of input number 2 at the instant the trigger is tripped. After the trigger is tripped, the element output will remain constant even if input signal number 1 falls below the prescribed trigger level. The trigger element is a two-input-single-output modeling element.

3.1.24 Jitter Element

The jitter element generates a random output uniformly distributed between 0 and 1. The jitter element is a zero-input-single-output modeling element.

3.1.25 Acceleration Element

The acceleration element computes the linear acceleration of a point on a rigid body expressed in the body axis system. This element is useful for modeling accelerometers with zero error. If it is assumed that $_I$ and $_B$ represent the inertial and body reference frames, then the acceleration of point A on the rigid body with respect to the inertial frame is given by equation 66 as follows:

$$\vec{\boldsymbol{a}}_{\boldsymbol{A/I}} = \vec{\boldsymbol{a}}_{\boldsymbol{C/I}} + \vec{\alpha}_{\boldsymbol{B/I}} \times \vec{\boldsymbol{r}}_{\boldsymbol{C->A}} + \vec{\omega}_{\boldsymbol{B/I}} \times \left(\vec{\omega}_{\boldsymbol{B/I}} \times \vec{\boldsymbol{r}}_{\boldsymbol{C->A}} \right). \tag{66}$$

In equation 66, $\vec{\omega}_{B/I}$ and $\vec{\alpha}_{B/I}$ are the angular velocity and acceleration vectors of the body, with respect to the inertial frame. Also, $\vec{r}_{C->A}$ is the position vector from point C to point A.

Equation 67 expresses equation 66 in the body frame, where point C is taken to be the mass center of the projectile as follows:

$$\begin{Bmatrix} a_{A_x} \\ a_{A_y} \\ a_{A_z} \end{Bmatrix} = \begin{Bmatrix} \dot{u} - rv + qw - (q^2 + r^2)\Delta_{SL} + (pq - \dot{r})\Delta_{BL} + (pr - \dot{q})\Delta_{WL} \\ \dot{v} - pw + ru + (pq + \dot{r})\Delta_{SL} - (p^2 + r^2)\Delta_{BL} + (qr - \dot{p})\Delta_{WL} \\ \dot{w} - qu + pv + (pr - \dot{q})\Delta_{SL} + (qr + \dot{p})\Delta_{BL} - (p^2 + q^2)\Delta_{WL} \end{Bmatrix}. \quad (67)$$

Equations 68–70 define the values p, q, r, u, v, and w used in equation 67 as follows:

$$\vec{\omega}_{B/I} = p\vec{i}_B + q\vec{j}_B + r\vec{k}_B, \quad (68)$$

$$\vec{\alpha}_{B/I} = \dot{p}\vec{i}_B + \dot{q}\vec{j}_B + \dot{r}\vec{k}_B, \quad (69)$$

$$\vec{v}_{C/I} = u\vec{i}_B + v\vec{j}_B + w\vec{k}_B, \quad (70)$$

and

$$\vec{r}_{C->A} = (SL_A - SL_{CG})\vec{i}_B + (BL_A - BL_{CG})\vec{j}_B + (BL_A - BL_{CG})\vec{k}_B. \quad (71)$$

In equation 71, SL, BL, and WL denote the stationline, buttline, and waterline of the projectile. The acceleration element is an 18-input-3-output modeling element. The order of the inputs and outputs to this element is shown in table 1.

Table 1. Acceleration input and output signals.

Input/Output	Signal
Input 1	U – I body component of the projectile mass center velocity
Input 2	V – J body component of the projectile mass center velocity
Input 3	W – K body component of the projectile mass center velocity
Input 4	P – I body component of the projectile angular velocity
Input 5	Q – J body component of the projectile angular velocity
Input 6	R – K body component of the projectile angular velocity
Input 7	U DOT – time derivative of U
Input 8	V DOT – time derivative of V
Input 9	W DOT – time derivative of W
Input 10	P DOT – time derivative of P
Input 11	Q DOT – time derivative of Q
Input 12	R DOT – time derivative of R
Input 13	SL A – stationline of point A
Input 14	BL A – buttline of point A
Input 15	WL A – waterline of point A
Input 16	SL CG – stationline of mass center of projectile
Input 17	BL CG – buttline of mass center of projectile
Input 18	WL CG – waterline of mass center of projectile
Output 1	I body component of acceleration
Output 2	J body component of acceleration
Output 3	K body component of acceleration

3.1.26 1D Table Lookup With Interpolation Element

Using two vectors of data, $\{x\}$ and $\{y\}$, the 1D table lookup with interpolation element generates an interpolated value, y^*, corresponding to the input value, x^*. If the input value, x^*, is out of the range of the table of data, then either the first or last element of the table is used, depending on the input point being out of range from above or below. The 1D table lookup with interpolation element is a single-input-single-output modeling element.

3.1.27 1D Table Lookup Without Interpolation Element

Using two vectors of data, $\{x\}$ and $\{y\}$, the 1D table lookup without interpolation element generates a value, y^*, corresponding to the input value, x^*. If the input value, x^*, is out of the range of the table of data, then either the first or last element of the table is used, depending on the input point being out of range from above or below. It should be noted that this element represents a non-interpolated table and simply generates y^* corresponding to the x value at or below the input value x^*. The 1D table lookup without interpolation element is a single-input-single-output modeling element.

3.1.28 Inertial/Body Euler Angle Transformation Element

The inertial/body transformation element transforms an input vector from the body frame to the inertial frame or vice versa. Euler angles are used to define the transformation between frames. Equations 72 and 73 provide the transformation equations for two transformation cases, inertial to body transformation using Euler angles and body to inertial transformation using Euler angles as follows:

$$\begin{Bmatrix} c_{X_B} \\ c_{Y_B} \\ c_{Z_B} \end{Bmatrix} = \begin{bmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ s_\phi s_\theta c_\psi - c_\phi s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi c_\theta \\ c_\phi s_\theta c_\psi + s_\phi s_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi & c_\phi c_\theta \end{bmatrix} \begin{Bmatrix} c_{X_I} \\ c_{Y_I} \\ c_{Z_I} \end{Bmatrix}, \tag{72}$$

and

$$\begin{Bmatrix} c_{X_I} \\ c_{Y_I} \\ c_{Z_I} \end{Bmatrix} = \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix} \begin{Bmatrix} c_{X_B} \\ c_{Y_B} \\ c_{Z_B} \end{Bmatrix}. \tag{73}$$

The body/inertial transformation element is a six-input-three-output element. The input/output order is shown in table 2.

Table 2. Inertial/body transformation (Euler angles) input and output signals.

Input/Output	Signal
Input 1	I component of vector
Input 2	J component of vector
Input 3	K component of vector
Input 4	PHI – Euler roll angle (rad)
Input 5	THETA – Euler pitch angle (rad)
Input 6	PSI – Euler yaw angle (rad)
Output 1	I body component of vector
Output 2	J body component of vector
Output 3	K body component of vector

3.1.29 Single-Axis Transformation Element

The single-axis transformation element transforms rotates an input vector about a prescribed angle to generate an output vector. There are six possible single-axis transformations that can be selected, shown in equations 74–79 as follows:

$$\begin{Bmatrix} \tilde{c}_X \\ \tilde{c}_Y \\ \tilde{c}_Z \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\eta & s_\eta \\ 0 & -s_\eta & c_\eta \end{bmatrix} \begin{Bmatrix} c_X \\ c_Y \\ c_Z \end{Bmatrix}, \tag{74}$$

$$\begin{Bmatrix} \tilde{c}_X \\ \tilde{c}_Y \\ \tilde{c}_Z \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\eta & -s_\eta \\ 0 & s_\eta & c_\eta \end{bmatrix} \begin{Bmatrix} c_X \\ c_Y \\ c_Z \end{Bmatrix}, \tag{75}$$

$$\begin{Bmatrix} \tilde{c}_X \\ \tilde{c}_Y \\ \tilde{c}_Z \end{Bmatrix} = \begin{bmatrix} c_\eta & 0 & -s_\eta \\ 0 & 1 & 0 \\ s_\eta & 0 & c_\eta \end{bmatrix} \begin{Bmatrix} c_X \\ c_Y \\ c_Z \end{Bmatrix}, \tag{76}$$

$$\begin{Bmatrix} \tilde{c}_X \\ \tilde{c}_Y \\ \tilde{c}_Z \end{Bmatrix} = \begin{bmatrix} c_\eta & 0 & s_\eta \\ 0 & 1 & 0 \\ -s_\eta & 0 & c_\eta \end{bmatrix} \begin{Bmatrix} c_X \\ c_Y \\ c_Z \end{Bmatrix}, \tag{77}$$

$$\begin{Bmatrix} \tilde{c}_X \\ \tilde{c}_Y \\ \tilde{c}_Z \end{Bmatrix} = \begin{bmatrix} c_\eta & s_\eta & 0 \\ -s_\eta & c_\eta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} c_X \\ c_Y \\ c_Z \end{Bmatrix}, \tag{78}$$

and

$$\begin{Bmatrix} \tilde{c}_X \\ \tilde{c}_Y \\ \tilde{c}_Z \end{Bmatrix} = \begin{bmatrix} c_\eta & -s_\eta & 0 \\ s_\eta & c_\eta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} c_X \\ c_Y \\ c_Z \end{Bmatrix}. \tag{79}$$

The single-axis transformation element is a four-input-three-output element. The input/output order is shown in table 3.

Table 3. Single-axis transformation input and output signals.

Input/Output	Signal
Input 1	I component of vector
Input 2	J component of vector
Input 3	K component of vector
Input 4	PHI – transformation rotation angle (rad)
Output 1	I body component of vector
Output 2	J body component of vector
Output 3	K body component of vector

3.1.30 Polynomial Filter Element

The polynomial filter element filters the input signal with a linear time invariant system that is described by a polynomial transfer function, shown in equation 80 as follows:

$$Y(s) = \frac{N_0 + N_1 s + ... + N_N s^N}{D_0 + D_1 s + ... + D_N s^N} U(s) \quad . \tag{80}$$

The polynomial filter element is a single-input-single-output element. The dynamic filter equations are integrated along with the rigid body equations of motion during a simulation.

3.1.31 State Space Filter Element

The state space filter element filters the input vector signal with a linear time invariant system that is described by state space matrices. The state space dynamic equations are shown in equations 81 and 82 as follows:

$$\{\dot{x}\} = [A]\{x\} + [B]\{u\}, \tag{81}$$

and

$$\{y\} = [C]\{x\} + [D]\{u\}. \tag{82}$$

The polynomial filter element is a multiple-input-multiple-output element. The dynamic filter equations are integrated along with the rigid body equations of motion during a simulation.

3.1.32 Sigmoid Element

Given an input signal, the sigmoid element computes the output of the sigmoid (or logistic) function. This element is represented mathematically by equation 83 as follows:

$$y_{FCS} = \frac{G}{1 - e^{\tau u_{FCS}}}. \tag{83}$$

Note that G is a user-defined gain, while τ is a user-defined decay constant. The sigmoid element is a single-input-single-output element.

3.1.33 AND Element

The AND element performs a logical AND operation on the inputs, producing a single output. This element can have up to 30 inputs. If all inputs equal 1, then the output is 1; otherwise, the output is 0. The AND element is a multiple-input-single-output element.

3.1.34 OR Element

The OR element performs a logical OR operation on the inputs, producing a single output. This element can have up to 30 inputs. If any input equals 1, then the output is 1. If all inputs equal 0, the output is 0. The OR element is a multiple-input-single-output element.

3.1.35 NOT Element

The NOT element toggles the input signal. If the input is ≤0.5, then the output is 1, while if the input is >0.5, the output is 0. The NOT element is a single-input-single-output element.

3.1.36 XOR Element

The XOR element performs a logical XOR operation on the inputs, producing a single output. This element can have up to 30 inputs. If all input values are 1 or if all input values are 0, the output is 0; otherwise, the output is 1.

3.1.37 Uniform Noise Element

The uniform noise element outputs a uniform noise signal defined by the input parameters. The uniform noise lies uniformly between the mean parameter plus/minus the range parameter. The uniform noise element is a zero-input-single-output modeling element. The input/output order is shown in table 4.

Table 4. Uniform noise input and output signals.

Input/Output	Signal
Input 1	Bias of uniform noise
Input 2	Range of uniform noise
Output	Noise signal

3.1.38 Gaussian Noise Element

The Gaussian noise element outputs a Gaussian noise signal defined by the mean and standard deviation input parameters. The Gaussian noise element is a zero-input-single-output modeling element. The input/output order is shown in table 5.

Table 5. Gaussian noise input and output signals.

Input/Output	Signal
Input 1	Mean of Gaussian noise
Input 2	Standard deviation of Gaussian noise
Output	Noise signal

3.1.39 Proportional Navigation Guidance Element

The proportional navigation guidance (PNG) element outputs the proportional navigation guidance command acceleration. Proportional navigation seeks to force the line of sight angle between the projectile and the target to be constant. Therefore, the acceleration command generated by PNG can be written as

$$\vec{A}_C = N_C \vec{V}_C \dot{\lambda}\,, \tag{84}$$

where $\vec{A}_C$ is the acceleration command, N_C is the PNG gain, $\vec{V}_C$ is the missile-target closing velocity, and λ is the line-of-sight angle. Let the *L* frame denote a reference frame with unit vector $\vec{I}_L$ aligned with the line of sight between the projectile and the target. Then, equation 84 can be expressed as

$$\vec{A}_C = -N_C \vec{v}_{C/I} \times \vec{\omega}_{L/I}\,. \tag{85}$$

In equation 85, $\vec{v}_{C/I}$ denotes the velocity of the projectile mass center with respect to the inertial frame. Also, noting that

$$\vec{\omega}_{L/I} = \frac{-\vec{r}_{C\to X} \times \vec{v}_{C/I}}{\left|\vec{r}_{C\to X}\right|^2}\,, \tag{86}$$

where $\vec{r}_{C\to X}$ denotes the distance vector from the projectile mass center to the target, the PNG-generated acceleration command can finally be written as

$$\vec{A}_C = \frac{N_C}{\left|\vec{r}_{C\to X}\right|^2} \vec{v}_{C/I} \times \left(\vec{r}_{C\to X} \times \vec{v}_{C/I}\right). \tag{87}$$

The inputs to the PNG element are the target's inertial frame position and velocity and the projectile's inertial frame position and velocity. In addition to command acceleration, the miss distance, closing velocity, and target Euler pitch and yaw angles are output as well. The PNG element is a 12-input-7-output modeling element. The input/output order is shown in table 6.

Table 6. PNG input and output signals.

Input/Output	Signal
Input 1	X inertial target position
Input 2	Y inertial target position
Input 3	Z inertial target position
Input 4	X inertial target velocity
Input 5	Y inertial target velocity
Input 6	Z inertial target velocity
Input 7	X inertial projectile position
Input 8	Y inertial projectile position
Input 9	Z inertial projectile position
Input 10	X inertial projectile velocity
Input 11	Y inertial projectile velocity
Input 12	Z inertial projectile velocity
Output 1	X inertial command acceleration
Output 2	Y inertial command acceleration
Output 3	Z inertial command acceleration
Output 4	Miss distance
Output 5	Closing velocity
Output 6	Target Euler pitch angle
Output 7	Target Euler yaw angle

3.1.40 Seeker Element

The seeker element simulates the outputs of a seeker. A seeker measures the projection of a target onto the seeker plane. The target in the seeker plane is typically defined by two angles, Γ_y and Γ_z. First, define the position of the target with respect to the seeker as

$$\begin{Bmatrix} e_x \\ e_y \\ e_z \end{Bmatrix} = [R][T_{IB}] \begin{Bmatrix} x_T - x \\ y_T - y \\ z_T - z \end{Bmatrix}, \tag{88}$$

where *R* is the transformation matrix from the projectile frame to the seeker frame. It should be noted that the $\vec{I}$ unit vector of the seeker frame lies along the seeker axis of symmetry. Then, the seeker angles are given by equations 89 and 90 as follows:

$$\Gamma_Y = \tan^{-1}\left(\frac{e_y}{e_x}\right) \quad , \tag{89}$$

and

$$\Gamma_Z = \tan^{-1}\left(\frac{e_z}{e_x}\right) \quad . \tag{90}$$

The seeker element determines whether the target is within the field of view by determining the total angle to the target and comparing it with the seeker field of view parameter, according to

$$\tan^{-1}\left(\frac{\sqrt{e_y^2+e_z^2}}{e_x}\right) < \Gamma_{FOV}\,. \tag{91}$$

User-defined parameters are the seeker's field of view (in radians), bias and standard deviation errors on the seeker output angles, and a transformation matrix relating the seeker sensor frame to the projectile body frame. The seeker requires the inertial position of the target as input, and it outputs a flag signifying if the target is within the seeker field of view, as well as line-of-sight angles to the target along the seeker frame *y* and *z* axes if the target is within the field of view. The seeker element is a three-input-three-output modeling element. The input/output order is shown in table 7.

Table 7. Seeker input and output signals.

Input/Output	Signal
Input 1	X inertial position of target
Input 2	Y inertial position of target
Input 3	Z inertial position of target
Output 1	Angle target makes with seeker Z axis, provided target is within field of view.
Output 2	Angle target makes with seeker Y axis, provided target is within field of view.
Output 3	Output is 0 if the target is out of the field of view and 1 if within the seeker field of view.

3.1.41 GPS Element

The GPS element simulates the output of an onboard GPS system. GPS errors are included through user-defined values for position and velocity bias errors, as well as variability in GPS position and velocity outputs expressed as a standard deviation. The GPS element outputs the projectile *x*, *y*, and *z* positions and velocities in the inertial frame. There are no inputs to the GPS element. The GPS element is a zero-input-six-output modeling element. The output order is shown in table 8.

Table 8. GPS output signals.

Output No.	Signal
1	X inertial position of projectile
2	Y inertial position of projectile
3	Z inertial position of projectile
4	X inertial velocity of projectile
5	Y inertial velocity of projectile
6	Z inertial velocity of projectile

3.1.42 Single-Axis Accelerometer Element

The single-axis accelerometer element simulates the output of a single-axis accelerometer. User-specified parameters are stationline, waterline, and buttline position of the accelerometer with respect to the projectile base, bias and standard deviation of accelerometer noise, scale factor, cross-axis sensitivities, and the transformation matrix relating the sensor frame to the projectile body frame. The element outputs the simulated accelerometer output. The single-axis accelerometer element is a zero-input-single-output modeling element.

3.1.43 Three-Axis Accelerometer Element

The three-axis accelerometer element simulates the output of a three-axis accelerometer. This element computes the acceleration felt by the accelerometer element in the same manner used for the acceleration element. Specifically, assuming that $_I$ and $_B$ represent the inertial and body reference frames, then the acceleration of point A on the rigid body with respect to the inertial frame is given by equation 92 as follows:

$$\vec{\boldsymbol{a}}_{\boldsymbol{A/I}} = \vec{\boldsymbol{a}}_{\boldsymbol{C/I}} + \vec{\alpha}_{\boldsymbol{B/I}} \times \vec{\boldsymbol{r}}_{\boldsymbol{C->A}} + \vec{\omega}_{\boldsymbol{B/I}} \times \left(\vec{\omega}_{\boldsymbol{B/I}} \times \vec{\boldsymbol{r}}_{\boldsymbol{C->A}} \right). \tag{92}$$

In equation 92, $\vec{\omega}_{B/I}$ and $\vec{\alpha}_{B/I}$ are the angular velocity and acceleration vectors of the body with respect to the inertial frame, and $\vec{r}_{C->A}$ is the position vector from point C to point A. Equation 93 expresses equation 92 in the body frame where point C is taken to be the mass center of the projectile as follows:

$$\begin{Bmatrix} \boldsymbol{a}_{A_x} \\ \boldsymbol{a}_{A_y} \\ \boldsymbol{a}_{A_z} \end{Bmatrix} = \begin{Bmatrix} \dot{\boldsymbol{u}} - \boldsymbol{rv} + \boldsymbol{qw} - (\boldsymbol{q}^2 + \boldsymbol{r}^2)\Delta_{SL} + (\boldsymbol{pq} - \dot{\boldsymbol{r}})\Delta_{BL} + (\boldsymbol{pr} - \dot{\boldsymbol{q}})\Delta_{WL} \\ \dot{\boldsymbol{v}} - \boldsymbol{pw} + \boldsymbol{ru} + (\boldsymbol{pq} + \dot{\boldsymbol{r}})\Delta_{SL} - (\boldsymbol{p}^2 + \boldsymbol{r}^2)\Delta_{BL} + (\boldsymbol{qr} - \dot{\boldsymbol{p}})\Delta_{WL} \\ \dot{\boldsymbol{w}} - \boldsymbol{qu} + \boldsymbol{pv} + (\boldsymbol{pr} - \dot{\boldsymbol{q}})\Delta_{SL} + (\boldsymbol{qr} + \dot{\boldsymbol{p}})\Delta_{BL} - (\boldsymbol{p}^2 + \boldsymbol{q}^2)\Delta_{WL} \end{Bmatrix}. \tag{93}$$

Equations 94–96 define the values p, q, r, u, v, and w used in equation 93 as follows:

$$\vec{\omega}_{\boldsymbol{B/I}} = \boldsymbol{p}\vec{\boldsymbol{i}}_{\boldsymbol{B}} + \boldsymbol{q}\vec{\boldsymbol{j}}_{\boldsymbol{B}} + \boldsymbol{r}\vec{\boldsymbol{k}}_{\boldsymbol{B}}, \tag{94}$$

$$\vec{\alpha}_{\boldsymbol{B/I}} = \dot{\boldsymbol{p}}\vec{\boldsymbol{i}}_{\boldsymbol{B}} + \dot{\boldsymbol{q}}\vec{\boldsymbol{j}}_{\boldsymbol{B}} + \dot{\boldsymbol{r}}\vec{\boldsymbol{k}}_{\boldsymbol{B}}, \tag{95}$$

and

$$\vec{\boldsymbol{v}}_{\boldsymbol{C/I}} = \boldsymbol{u}\vec{\boldsymbol{i}}_{\boldsymbol{B}} + \boldsymbol{v}\vec{\boldsymbol{j}}_{\boldsymbol{B}} + \boldsymbol{w}\vec{\boldsymbol{k}}_{\boldsymbol{B}}. \tag{96}$$

In equation 97, SL, BL, WL denote the stationline, buttline, and waterline of the projectile. The user can specify the stationline, waterline, and buttline of the element with respect to the projectile base, as well as bias and standard deviation of accelerometer noise, scale factor, cross-axis sensitivities for all three axes.

$$\vec{\boldsymbol{r}}_{\boldsymbol{C->A}} = (\boldsymbol{SL}_{\boldsymbol{A}} - \boldsymbol{SL}_{\boldsymbol{CG}})\vec{\boldsymbol{i}}_{\boldsymbol{B}} + (\boldsymbol{BL}_{\boldsymbol{A}} - \boldsymbol{BL}_{\boldsymbol{CG}})\vec{\boldsymbol{j}}_{\boldsymbol{B}} + (\boldsymbol{BL}_{\boldsymbol{A}} - \boldsymbol{BL}_{\boldsymbol{CG}})\vec{\boldsymbol{k}}_{\boldsymbol{B}}. \tag{97}$$

Another user-defined parameter is the transformation matrix relating the sensor frame to the projectile body frame. The element outputs simulated accelerometer outputs along all three axes. The three-axis accelerometer element is a zero-input-three-output modeling element. The output order is shown in table 9.

Table 9. Three-axis accelerometer output signals.

Output No.	Signal
1	Accelerometer reading along the X sensor axis
2	Accelerometer reading along the Y sensor axis
3	Accelerometer reading along the Z sensor axis

3.1.44 Single-Axis Magnetometer Element

The single-axis magnetometer element simulates the output of a single-axis magnetometer. A magnetometer measures the inner product between the magnetometer's sensitive axis and the Earth's magnetic field. Since magnetometers do not directly measure any projectile states, processing is required to obtain useful sensor feedback data. Given the Earth's magnetic field in Earth-fixed coordinates,

$$\vec{m} = m_X \vec{I}_I + m_Y \vec{J}_I + m_Z \vec{K}_I \cdot \tag{98}$$

The single-axis magnetometer measures

$$\vec{\boldsymbol{m}}_{\boldsymbol{MAG}} = \vec{\boldsymbol{m}} \bullet \vec{\boldsymbol{I}}_{\boldsymbol{MAG}}, \tag{99}$$

where $\vec{\boldsymbol{I}}_{\boldsymbol{MAG}}$ is the unit vector along the magnetometer's sensitive axis. The user must specify the inertial frame *x*, *y*, and *z* component of the Earth's magnetic field unit vector. In addition, the user can specify bias and standard deviation of sensor noise, scale factor, cross-axis sensitivities, and the transformation matrix relating the sensor frame to the projectile body frame. The element outputs a single value representing the magnetometer output. The single-axis magnetometer is a zero-input-single-output modeling element.

3.1.45 Three-Axis Magnetometer Element

The three-axis magnetometer element simulates the output of a three-axis magnetometer. Given the Earth's magnetic field in Earth-fixed coordinates,

$$\vec{m} = m_X \vec{I}_I + m_Y \vec{J}_I + m_Z \vec{K}_I \cdot \tag{100}$$

The three-axis magnetometer measures this vector in the sensor reference frame, given as

$$\vec{m} = \tilde{m}_X \vec{I}_S + \tilde{m}_Y \vec{J}_S + \tilde{m}_Z \vec{K}_S \cdot \tag{101}$$

Equating components,

$$\begin{Bmatrix} \tilde{m}_X \\ \tilde{m}_Y \\ \tilde{m}_Z \end{Bmatrix} = [T_S][T_B] \begin{Bmatrix} m_X \\ m_Y \\ m_Z \end{Bmatrix}, \tag{102}$$

where $\boldsymbol{T_B}$ is the standard inertial-to-body-frame transformation, and $\boldsymbol{T_S}$ is the body-to-sensor-frame transformation. The user must specify the inertial frame *x*, *y*, and *z* component of the Earth's magnetic field unit vector. The user can also specify bias and standard deviation of sensor noise, scale factor, and cross-axis sensitivities for all three axes. The transformation matrix relating the sensor frame to the projectile body frame must also be defined. The element outputs the simulated magnetometer output along all three axes. The three-axis magnetometer element is a zero-input-three-output modeling element. The output order is shown in table 10.

Table 10. Three-axis magnetometer output signals.

Output No.	Signal
1	Magnetometer reading along the X sensor axis
2	Magnetometer reading along the Y sensor axis
3	Magnetometer reading along the Z sensor axis

3.1.46 Single-Axis Rate Gyroscope Element

The single-axis gyroscope element simulates the output of a single-axis gyroscope. The user can specify bias and standard deviation of sensor noise, scale factor, cross-axis sensitivities, and the transformation matrix relating the sensor frame to the projectile body frame. The element outputs projectile angular rate along the gyroscope axis. The single-axis gyroscope element is a zero-input-single-output modeling element.

3.1.47 Three-Axis Rate Gyroscope Element

The three-axis gyroscope element simulates the output of a three-axis gyroscope. The user can specify bias and standard deviation of sensor noise, scale factor, and cross-axis sensitivities for all three axes. The transformation matrix relating the sensor frame to the projectile body frame must also be defined. The element outputs projectile angular rate along all three sensor axes. The three-axis gyroscope element is a zero-input-three-output modeling element. The output order is shown in table 11.

Table 11. Three-axis gyroscope output signals.

Output No.	Signal
1	Gyroscopic reading along the X sensor axis
2	Gyroscopic reading along the Y sensor axis
3	Gyroscopic reading along the Z sensor axis

3.1.48 Solar Sensor

The solar sensor element generates a pulse train that simulates the output of a solar sensor aimed out a port in the side of the projectile. Solar sensors output 1 when light impacts the sensor and 0 when the light level is below a certain threshold. The sensor is typically placed under a narrow slit within the projectile. The slit configuration is defined by two angles (γ_{IJ} in the XY plane of the sensor frame and γ_{IK} in the XZ plane of the sensor frame), which determine the range of angles a vector can make with the sensor and not be blocked by the slit. Furthermore, it should be assumed that the sun is at a certain location in the sky with respect to the inertial frame, defined by the angles ψ_{Sun} and θ_{Sun}. Then, the sun reference frame can be expressed as follows:

$$\begin{Bmatrix} \vec{I}_{Sun} \\ \vec{J}_{Sun} \\ \vec{K}_{Sun} \end{Bmatrix} = \begin{bmatrix} c_{\theta_{Sun}} & 0 & -s_{\theta_{Sun}} \\ 0 & 1 & 0 \\ s_{\theta_{Sun}} & 0 & c_{\theta_{Sun}} \end{bmatrix} \begin{bmatrix} c_{\psi_{Sun}} & s_{\psi_{Sun}} & 0 \\ -s_{\psi_{Sun}} & c_{\psi_{Sun}} & 0 \\ 0 & 0 & 1 \end{bmatrix} [T_B]^T [T_S]^T \begin{Bmatrix} \vec{I}_S \\ \vec{J}_S \\ \vec{K}_S \end{Bmatrix} = [T] \begin{Bmatrix} \vec{I}_S \\ \vec{J}_S \\ \vec{K}_S \end{Bmatrix}, \qquad (103)$$

where T_B is the standard inertial to body frame transformation, and T_S is the body to sensor frame transformation. To determine if the solar sensor reads 1 or 0, the vector $\vec{I}_{Sun}$ must impact the solar sensor through the slit, i.e., within $\pm\gamma_{IJ}$ and $\pm\gamma_{IK}$. This check is accomplished by the following logic in equation 104:

$$\begin{aligned} &\text{If } T(1,1) < 0 \rightarrow \text{Sensor Reads 0} \\ &\text{If } \begin{cases} \tan^{-1}\left(\left|T(1,2)\right|, T(1,1)\right) < \gamma_{IJ} \\ \tan^{-1}\left(\left|T(1,3)\right|, T(1,1)\right) < \gamma_{IK} \end{cases} \rightarrow \text{Sensor Reads 1} \cdot \\ &\text{Otherwise Sensor Reads 0} \end{aligned} \qquad (104)$$

User-specified parameters are the solar azimuth and elevation angles and the transformation matrix relating the sensor frame to the projectile body frame. The solar sensor element is a zero-input-single-output modeling element.

3.1.49 Inertial Measurement Unit Element

The inertial measurement unit (IMU) element simulates the output of an inertial measurement unit. A typical IMU system uses angular velocity inputs from a three-axis rate gyro and acceleration inputs from a three-axis accelerometer to determine projectile velocity, position, and orientation. Time derivatives of velocities are computed using accelerometer and angular velocity output using the following expression:

$$\begin{Bmatrix} \dot{u}_{IMU} \\ \dot{v}_{IMU} \\ \dot{w}_{IMU} \end{Bmatrix} = \begin{Bmatrix} a_{XIMU} \\ a_{YIMU} \\ a_{ZIMU} \end{Bmatrix} - \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \begin{Bmatrix} u_{IMU} \\ v_{IMU} \\ w_{IMU} \end{Bmatrix}. \tag{105}$$

Likewise, derivatives of Euler angles and position states are calculated using equations 106 and 107 as follows:

$$\begin{Bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Bmatrix} = \begin{bmatrix} 1 & \boldsymbol{s}_{\phi}\boldsymbol{t}_{\theta} & \boldsymbol{c}_{\phi}\boldsymbol{t}_{\theta} \\ 0 & \boldsymbol{c}_{\phi} & -\boldsymbol{s}_{\phi} \\ 0 & \boldsymbol{s}_{\phi}/\boldsymbol{c}_{\theta} & \boldsymbol{c}_{\phi}/\boldsymbol{c}_{\theta} \end{bmatrix} \begin{Bmatrix} \boldsymbol{p} \\ \boldsymbol{q} \\ \boldsymbol{r} \end{Bmatrix}, \tag{106}$$

and

$$\begin{Bmatrix} \dot{\boldsymbol{x}}_{\boldsymbol{IMU}} \\ \dot{\boldsymbol{y}}_{\boldsymbol{IMU}} \\ \dot{\boldsymbol{z}}_{\boldsymbol{IMU}} \end{Bmatrix} = \begin{bmatrix} \boldsymbol{c}_{\theta}\boldsymbol{c}_{\psi} & \boldsymbol{s}_{\phi}\boldsymbol{s}_{\theta}\boldsymbol{c}_{\psi} - \boldsymbol{c}_{\phi}\boldsymbol{s}_{\psi} & \boldsymbol{c}_{\phi}\boldsymbol{s}_{\theta}\boldsymbol{c}_{\psi} - \boldsymbol{s}_{\phi}\boldsymbol{s}_{\psi} \\ \boldsymbol{c}_{\theta}\boldsymbol{s}_{\psi} & \boldsymbol{s}_{\phi}\boldsymbol{s}_{\theta}\boldsymbol{s}_{\psi} - \boldsymbol{c}_{\phi}\boldsymbol{c}_{\psi} & \boldsymbol{c}_{\phi}\boldsymbol{s}_{\theta}\boldsymbol{s}_{\psi} - \boldsymbol{s}_{\phi}\boldsymbol{c}_{\psi} \\ -\boldsymbol{s}_{\theta} & \boldsymbol{s}_{\phi}\boldsymbol{c}_{\theta} & \boldsymbol{c}_{\phi}\boldsymbol{c}_{\theta} \end{bmatrix} \begin{Bmatrix} \boldsymbol{u}_{\boldsymbol{IMU}} \\ \boldsymbol{v}_{\boldsymbol{IMU}} \\ \boldsymbol{w}_{\boldsymbol{IMU}} \end{Bmatrix}. \tag{107}$$

Body velocities, Euler angles, and position states are updated by integrating equations 105–107. The IMU element incorporates both accelerometer and gyroscope noise. User-defined parameters for the IMU element are listed in order in table 12, while IMU outputs are listed in order in table 13. The IMU element is a zero-input-24-output modeling element.

Table 12. IMU element user-defined input parameters.

Parameter	Input Parameters
1	Stationline of accelerometer on IMU
2	Buttline of accelerometer on IMU
3	Waterline of accelerometer on IMU
4	AX accelerometer bias
5	AY accelerometer bias
6	AZ accelerometer bias
7	AX accelerometer noise standard deviation
8	AY accelerometer noise standard deviation
9	AZ accelerometer noise standard deviation
10	AX accelerometer scale factor
11	AX-Y cross-axis sensitivity
12	AX-Z cross-axis sensitivity
13	AY-X cross-axis sensitivity
14	AY accelerometer scale factor
15	AY-Z cross-axis sensitivity
16	AZ-X cross-axis sensitivity
17	AZ-Y cross-axis sensitivity
18	AZ accelerometer scale factor
19	X bias of gyroscope reading
20	Y bias of gyroscope reading
21	Z bias of gyroscope reading
22	X noise standard deviation of gyroscope reading
23	Y noise standard deviation of gyroscope reading
24	Z noise standard deviation of gyroscope reading
25	Scale factor of ω_x output
26	Cross-axis sensitivity of ω_x to ω_y
27	Cross-axis sensitivity of ω_x to ω_z
28	Cross-axis sensitivity of ω_y to ω_x
29	Scale factor of ω_y output
30	Cross-axis sensitivity of ω_y to ω_z
31	Cross-axis sensitivity of ω_z to ω_x
32	Cross-axis sensitivity of ω_z to ω_y
33	Scale factor of ω_z output
34	T(1,1) transformation matrix element relating sensor frame to the projectile body frame
35	T(1,2) transformation matrix element relating sensor frame to the projectile body frame
36	T(1,3) transformation matrix element relating sensor frame to the projectile body frame
37	T(2,1) transformation matrix element relating sensor frame to the projectile body frame
38	T(2,2) transformation matrix element relating sensor frame to the projectile body frame
39	T(2,3) transformation matrix element relating sensor frame to the projectile body frame
40	T(3,1) transformation matrix element relating sensor frame to the projectile body frame
41	T(3,2) transformation matrix element relating sensor frame to the projectile body frame
42	T(3,3) transformation matrix element relating sensor frame to the projectile body frame

Table 13. IMU output signals.

Output No.	Signal
1	X – I inertial component of projectile mass center position
2	Y – J inertial component of projectile mass center position
3	Z – K inertial component of projectile mass center position
4	PHI Euler roll angle
5	THETA Euler pitch angle
6	PSI Euler yaw angle
7	U – I body component of the projectile mass center velocity
8	V – J body component of the projectile mass center velocity
9	W – K body component of the projectile mass center velocity
10	X DOT – time derivative of X
11	Y DOT – time derivative of Y
12	Z DOT – time derivative of Z
13	PHI DOT – time derivative of PHI
14	THETA DOT – time derivative of THETA
15	PSI DOT – time derivative of PSI
16	U DOT – time derivative of U
17	V DOT – time derivative of V
18	W DOT – time derivative of W
19	AX accelerometer output
20	AY accelerometer output
21	AZ accelerometer output
22	P gyroscope output
23	Q gyroscope output
24	R gyroscope output

3.1.50 Positive Crossing Element

The positive crossing element outputs the number of times an input signal has crossed a threshold value in the positive direction (from below the threshold to above the threshold). The user can specify an arbitrary number of input signals and one threshold value. The element outputs the signal value at the occurrence of positive crossing for each signal, as well as a counter that keeps track of positive crossings. The positive crossing element requires *N* inputs and provides *N*+1 outputs. The input/output order is shown in table 14.

Table 14. Positive crossing output signals.

Input/Output	Signal
Input 1	Signal 1
…Input N	Signal N
Output 1	Signal 1 at positive crossing
…Output N	Signal N at positive crossing
Output N+1	Number of positive crossings

3.1.51 Negative Crossing Element

The negative crossing element outputs the number of times an input signal has crossed a threshold value in the negative direction (from above the threshold to below the threshold). The user can specify an arbitrary number of input signals and one threshold value. The element outputs the signal value at the occurrence of negative crossing for each signal, as well as a counter that keeps track of negative crossings. The negative crossing element requires *N* inputs and provides *N*+1 outputs. The input/output order is shown in table 15.

Table 15. Negative crossing output signals.

Input/Output	Signal
Input 1	Signal 1
…Input N	Signal N
Output 1	Signal 1 at negative crossing
…Output N	Signal N at negative crossing
Output N+1	Number of negative crossings

3.1.52 Zero Order Hold Element

The zero order hold element holds an input signal for a specified amount of time, resampling it along discrete intervals referred to as the hold time. Figure 4 demonstrates this concept. Given an input signal and a hold time of 1 s, the held input signal is computed by sampling the input, holding the value for 1 s, and then resampling it again.

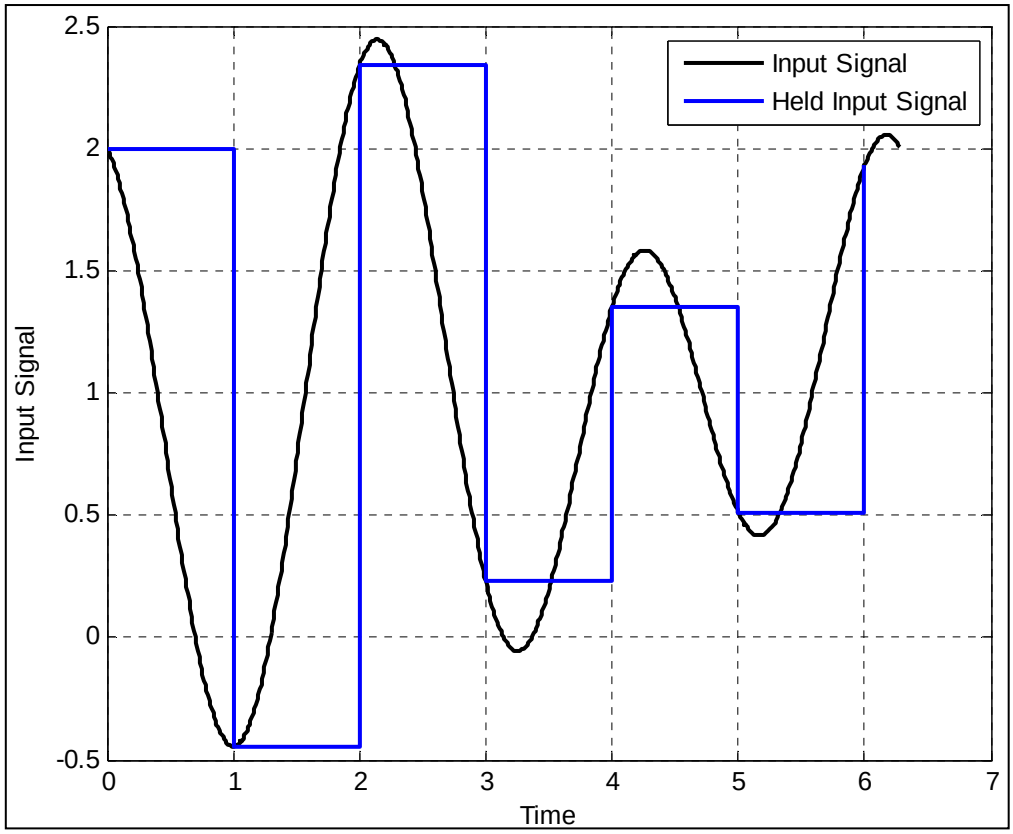

Figure 4. Zero order hold example output.

The element requires the user to specify the hold time. The output of the element is the held input signal. The zero order hold element is a single-input-single-output element.

3.2 Example Control System Diagram

This section describes a notional control system designed to drive the projectile roll angle to zero. Figure 5 shows a block diagram of this example controller.

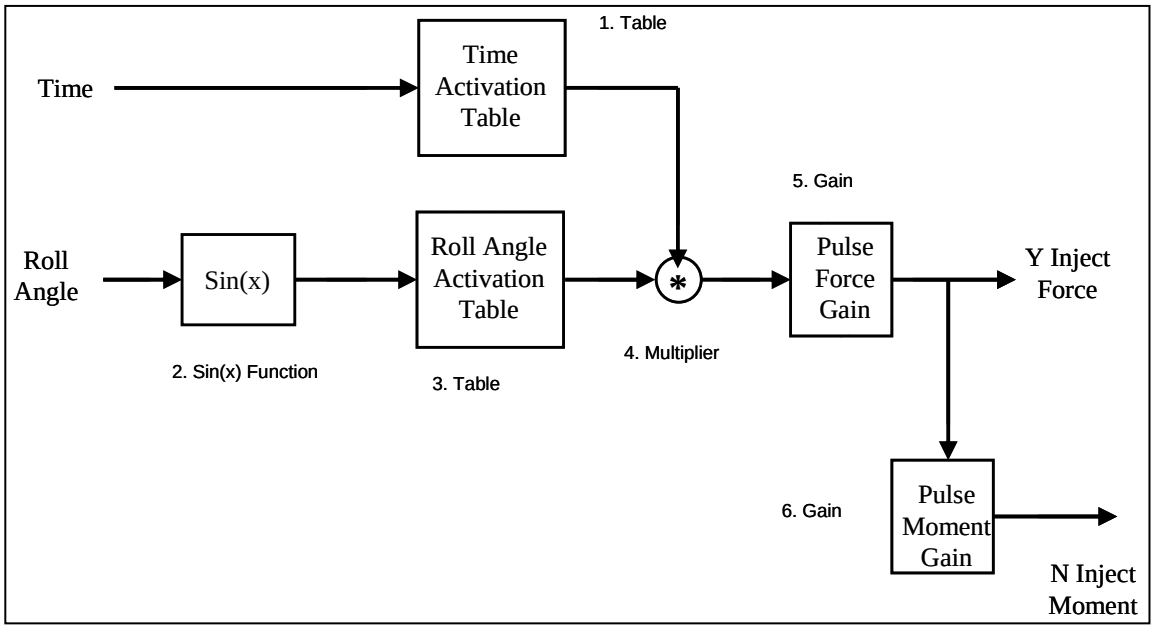

Figure 5. Example block diagram.

This controller uses sine, gain, multiply, and 1D table with interpolation elements to compute injection forces and moments. Each modeling element in the diagram has a corresponding index. The outputs from the flight control system are the *y* injection force and *N* injection moment.

4. Running Boom

BOOM is run like any other computer program. For example, on a PC or MAC, a user can simply double click on the BOOM executable and the program will begin to run. When BOOM is run, it first searches for a file called 'BOOM.ifiles' in the same directory as the program. The file 'BOOM.ifiles' is a file of input data file names, which must include the entire input data file path (see appendix H). The code requires body aerodynamics, mass properties, atmosphere, initial conditions, and configuration input files in order to run, as described in appendices A–E. Controlled flight simulations require a control system file. Canard data are input through a canard file, as outlined in appendix F. After each simulation, BOOM outputs both a log file and a time history of the projectile states, as outlined in appendix G. Dispersion simulations require the use of a dispersion input file, shown in appendix I, and output a dispersion output file described in appendix J. It should be noted that all example files are provided simply to demonstrate file formats and do not represent actual data for any particular projectile.

5. Conclusion

This report outlines the theory and implementation of the BOOM computer-aided smart weapons simulation tool. The computer-simulation program uses a 6-DOF flight simulation model coupled with an open-structure flight control modeling system, providing weapons designers with a valuable tool for design evaluation, performance prediction, and control system development. A wide variety of control system blocks are implemented within BOOM, and each is described in detail. Future reports will provide example control system files and example trajectory simulations.

Appendix A. .BODY Input File

The .BODY input file lists the aerodynamic coefficients for the projectile, which vary as a function of Mach number. The .BODY file for the example projectile is given as follows:

```
3.0000000000e-001  ! Reference Diameter for Aerodynamic Loads (ft)
4.0000000000e+000  ! Number of Body Fins (nd)
1.2000000000e+001  ! Number of Mach Number Points (nd)
4.0000000000e-001  ! Mach Points (nd)
6.0000000000e-001
7.0000000000e-001
7.5000000000e-001
8.0000000000e-001
8.5000000000e-001
8.7500000000e-001
9.0000000000e-001
9.2500000000e-001
9.5000000000e-001
9.7500000000e-001
1.0000000000e+000
3.00000000000e-001  ! SLCOP (ft)
5.00000000000-001
5.00000000000-001
5.00000000000-001
5.00000000000-001
5.00000000000-001
5.00000000000-001
5.00000000000-001
5.00000000000-001
5.00000000000-001
5.00000000000-001
0.0000000000e+000  ! BLCOP (ft)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000  ! WLCOP (ft)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000  ! SLMAG (ft)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
```

```
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000  ! BLMAG (ft)
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000  ! WLMAG (ft)
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000  ! SLMAGA2 (ft)
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000  ! BLMAGA2 (ft)
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000  ! WLMAGA2 (ft)
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000  ! SLMAGA4 (ft)
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
0.00000000000e+000
```

```
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000  ! BLMAGA4 (ft)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000  ! WLMAGA4 (ft)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
1.000000000e-001  ! CX0 (nd)
1.0000000000e-001
1.0000000000e-001
1.0000000000e-001
1.0000000000e-001
1.0000000000e-001
1.0000000000e-001
2.0000000000e-001
2.0000000000e-001
2.0000000000e-001
3.0000000000e-001
4.0000000000e-001
1.0000000000e-001  ! CX0BB (nd)
1.000000000e-001
1.0000000000e-001
1.0000000000e-001
1.0000000000e-001
1.0000000000e-001
1.0000000000e-001
2.0000000000e-001
2.0000000000e-001
2.0000000000e-001
3.0000000000e-001
4.0000000000e-001
1.0000000000e+000  ! CX2 (nd)
1.0000000000e+000
1.0000000000e+000
1.0000000000e+000
1.0000000000e+000
2.0000000000e+000
2.0000000000e+000
2.0000000000e+000
2.0000000000e+000
2.0000000000e+000
3.0000000000e+000
3.0000000000e+000
5.0000000000e+000  ! CNA (nd)
6.0000000000e+000
6.0000000000e+000
7.0000000000e+000
7.0000000000e+000
```

```
7.0000000000e+000
7.0000000000e+000
7.0000000000e+000
8.0000000000e+000
8.0000000000e+000
8.0000000000e+000
9.0000000000e+000
0.0000000000e+000  ! CNA3 (nd)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000  ! CNGAMA3 (nd)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000  ! CY0 (nd)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000  ! CZ0 (nd)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000  ! CYPA (nd)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
3.000000000e+000  ! CLP (nd)
3.0000000000e+000
3.0000000000e+000
3.0000000000e+000
```

```
3.0000000000e+000
4.0000000000e+000
4.0000000000e+000
4.0000000000e+000
4.0000000000e+000
4.0000000000e+000
4.0000000000e+000
4.0000000000e+000
3.000000000000e-002  ! CLDD (nd)
4.0000000000e-002
4.0000000000e-002
4.0000000000e-002
4.0000000000e-002
4.0000000000e-002
5.0000000000e-002
5.0000000000e-002
5.0000000000e-002
5.0000000000e-002
5.0000000000e-002
5.0000000000e-002
0.0000000000e+000  ! CLGAMA3 (nd)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
7.0000000000e+001  ! CMQ (nd)
8.0000000000e+001
8.0000000000e+001
8.0000000000e+001
8.0000000000e+001
9.0000000000e+001
9.0000000000e+001
9.0000000000e+001
9.0000000000e+001
9.0000000000e+001
9.0000000000e+001
1.0000000000e+002
0.0000000000e+000  ! CMQA2 (nd)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
```

INTENTIONALLY LEFT BLANK.

Appendix B. .TIME File

The .TIME file describes certain key simulation parameters, such as timestep and final simulation time, among others. It also sets initial conditions for the simulation. The example projectile .TIME file is given as follows:

```
     0.00000     ! initial time (sec), TINITIAL
    43.0000      ! final integration time (sec), TFINAL
     0.010     ! integration time step, (sec), DT
    1000         ! time history output skip parameter (nd),ITOUTSKIP
    18           ! number of misc. simulation outputs (nd), NOTHER
     0.00000     ! initial x inertial position (ft), x
     0.00000     ! initial y inertial position (ft), y
     0.00000     ! initial z inertial position (ft), z
     0.00000     ! initial roll angle (rad), phi
     0.2000      ! initial pitch angle (rad), theta
     0.000000    ! initial yaw angle (rad), psi
  100.000      ! initial x body velocity (ft/s), u
     0.0000      ! initial y body velocity (ft/s), v
     0.0000      ! initial z body velocity (ft/s), w
     0.0000      ! initial roll rate (r/s), p
     0.0000      ! initial pitch rate (r/s), q
     0.0000      ! initial yaw rate (r/s), r
```

INTENTIONALLY LEFT BLANK.

Appendix C. .MI File

The .MI file contains the mass properties of the projectile. The example projectile .MI file for the controlled trajectory is given as follows:

```
 1.00000     ! total weight of projectile (lbf)
0.2900      ! stationline of c.g. (ft)
0.0000        ! buttline of c.g. (ft)
0.0000        ! waterline of c.g. (ft)
0.007    ! ixx moment of inertia (slug*ft^2)
0.0500      ! iyy moment of inertia (slug*ft^2)
0.0500      ! izz moment of inertia (slug*ft^2)
0.0000        ! ixy moment of inertia (slug*ft^2)
0.0000        ! ixz moment of inertia (slug*ft^2)
0.0000        ! iyz moment of inertia (slug*ft^2)
1             ! body aerodynamic flag, (0=off,1=on)
1             ! gravity flag, (0=off,1=on)
1             ! canards flag, (0=off,1=on)
0             ! force injection flag, (0=off,1=on)
0             ! rocket assist flag, (0=off,1=on)
```

INTENTIONALLY LEFT BLANK.

Appendix D. .ATM File

The .ATM file lists atmospheric wind parameters and allows for atmosphere model selection. An example .ATM file is given as follows:

```
0.000000     ! atmospheric wind intensity (m/s), SIGMA
0.000000     ! atmospheric wind azimuthal angle (rad), PSIWIND
1            ! atmospheric density flag (0=constant,1=eqn,2=table)
```

INTENTIONALLY LEFT BLANK.

Appendix E. .CONFIG File

The .CONFIG file allows the user to activate or deactivate the flight controller, select the physical projectile model type, and select between a single time simulation or a dispersion simulation. An example .CONFIG file for a single controlled simulation is shown as follows:

```
1.0          ! physical projectile model type (nd), IBMODEL
1.0          ! fcs configuration (0=off,1=on), IFCS
0.0          ! analysis type (0=time,1=dispersion), IANALTYPE
```

INTENTIONALLY LEFT BLANK.

Appendix F. .CAN File

The .CAN file describes each individual canard and provides canard aerodynamic coefficients. An example .CAN file for the example projectile is given as follows:

```
2.0000000000e+000  ! Number of Canards (nd)
0.001000000000000  ! 1: Reference Area (ft^2), + J Body Canard, Maneuver
1.000000000000000  ! 1: Stationline (ft)
0.000000000000000  ! 1: Buttline (ft)
0.0000000000e+000  ! 1: Waterline (ft)
0.0000000000e+000  ! 1: Sweep Angle (rad)
0.0000000000e+000  ! 1: Roll Angle (rad)
1.0000000000e-001  ! 1: Pitch Angle (rad)
0.001000000000000  ! 2: Reference Area (ft^2), - J Body Canard, Maneuver
1.000000000000000  ! 2: Stationline (ft)
0.000000000000000  ! 2: Buttline (ft)
0.0000000000e+000  ! 2: Waterline (ft)
0.0000000000e+000  ! 2: Sweep Angle (rad)
3.1415926530e+000  ! 2: Roll Angle (rad)
-1.000000000e-001 ! 2: Pitch Angle (rad)
2.4000000000e+001  ! Number of Mach Points (nd)
1.0000000000e-002  ! Mach Points (nd)
4.0000000000e-001
6.0000000000e-001
7.0000000000e-001
7.5000000000e-001
8.0000000000e-001
8.5000000000e-001
8.7500000000e-001
9.0000000000e-001
9.2500000000e-001
9.5000000000e-001
9.7500000000e-001
1.0000000000e+000
1.0250000000e+000
1.0500000000e+000
1.1000000000e+000
1.2000000000e+000
1.3500000000e+000
1.5000000000e+000
1.7500000000e+000
2.0000000000e+000
2.2500000000e+000
2.5000000000e+000
3.0000000000e+000
6.0000000000e-000  ! CL Alpha (nd)
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
```

```
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
6.0000000000e-000
0.0000000000e+000  ! CL Alpha^3 (nd)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000  ! CL Alpha^5 (nd)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0018250000000000  ! CD0 (nd)
0.0018750000000000
0.0019500000000000
0.0021250000000000
0.0022500000000000
0.0023250000000000
0.0024500000000000
0.0025000000000000
0.0025750000000000
0.0026250000000000
0.0026250000000000
0.0021500000000000
0.0037750000000000
0.0039500000000000
0.0034500000000000
0.0041250000000000
0.0075000000000000
```

```
0.008250000000000
0.007125000000000
0.007850000000000
0.008125000000000
0.006700000000000
0.005950000000000
0.006300000000000
0.109800000000000  ! CD2 (nd)
0.120000000000000
0.125200000000000
0.126100000000000
0.126500000000000
0.126900000000000
0.129900000000000
0.130800000000000
0.132900000000000
0.135000000000000
0.135800000000000
0.136500000000000
0.134500000000000
0.132900000000000
0.130100000000000
0.129800000000000
0.126200000000000
0.112800000000000
0.108300000000000
0.106800000000000
0.098700000000000
0.093200000000000
0.089600000000000
0.080500000000000
0.0000000000e+000  ! Ci (nd)
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
0.0000000000e+000
```

INTENTIONALLY LEFT BLANK.

Appendix G. _BODY.out File

The _BODY.out file provides a time history of each state variable in columns according to the following format: *time, x, y, z, ϕ, θ, ψ, u, v, w, p, q, r*.

INTENTIONALLY LEFT BLANK.

Appendix H. BOOM.ifiles File

All input filenames are listed in the BOOM.ifiles file. Files named “empty.xxx” are disregarded. An example BOOM.ifiles file is given as follows:

```
Example.CONFIG
Example.ATM
Example.MI
Example.BODY
Example.CAN
empty.rap
ExampleControl.CS
Example.TIME
empty.dis
Example_BODY.out
Example_FCS.out
Example_MISC.out
Example_DIS.out
Example.log
```

INTENTIONALLY LEFT BLANK.

Appendix I. .DIS File

The .DIS input file contains several flags relating to the dispersion simulation, as well as actual variable perturbation values. The *dispersion type* flag selects between horizontal and vertical impact dispersion simulations used for indirect fire and direct fire, respectively. The *dispersion impact value* allows the user to choose the distance at which the projectile hits the target, corresponding to the height of the ground at the impact range for indirect fire or the downrange distance to target for direct fire. The user then selects the number of Monte Carlo simulations and the number of dispersion parameters. Dispersion parameters are listed with their global variable names (i.e., 'XBODYIC(7)' refers to initial value of state 7, or initial muzzle velocity) and the perturbation from the mean value, given in the .TIME file for body initial conditions and the .ATM file for atmospheric wind conditions. An example .DIS file is shown as follows:

```
1          ! dispersion type (nd, 0=vertical plane, 1=horizontal plane)
0.0000001       ! dispersion impact value (ft)
20              ! number of dispersion points (nd)
7               ! number of dispersion parameters (nd)
'XBODYIC(7)'    ! 1: dispersion variable, initial muzzle velocity (ft/sec)
       -3.6768
      -14.1575
        1.0653
        2.4452
       -9.7450
       10.1228
       10.1079
       -0.3199
        2.7820
        1.4844
       -1.5870
        6.1692
       -5.0007
       18.5571
       -1.1594
        0.9684
        9.0675
        0.5039
       -0.8130
       -7.0750
   'XBODYIC(5)'     ! 2: dispersion variable, initial pitch angle (rad)
        0.0111
       -0.0105
        0.0096
       -0.0192
        0.0015
       -0.0350
       -0.0086
        0.0081
       -0.0056
```

```
     0.0216
    -0.0110
    -0.0406
    -0.0215
     0.0184
    -0.0020
     0.0066
     0.0165
    -0.0370
    -0.0113
    -0.0123
'XBODYIC(6)'       ! 3: dispersion variable, initial yaw angle (rad)
     0.0132329574034403
     -0.012076634444053
     0.0118713552353676
    -0.0187193723309365
      0.015703985414758
    -0.0370549527182541
    0.00496916661034146
    -0.0127988996113377
    -0.0134979510830484
    0.00265013800811126
   -0.00587902163686497
     0.0169429707734835
   -0.00187162751464533
     0.0176887702351092
   -0.00829636601931578
    0.00120337214420554
    0.00695674395727813
     0.0194835683322056
     0.0108289149933971
   -0.00502086691678002
'XBODYIC(8)'            ! 4: dispersion variable, initial v (ft/sec)
       2.74025479345764
       3.72464706141741
       3.32071227433058
      -1.81942141053491
       1.06239057975022
       2.18663496321023
      -2.57780548137631
      0.484228582486091
       1.83087173554233
      -3.09164927381371
      -2.28710837911641
        1.8676188012438
    -0.0151945669438773
       1.25876834296447
       3.27425230901719
       1.15689439317901
      -1.88895802804802
       1.80479823869999
     -0.400530984074188
      -1.95894746534556
'XBODYIC(9)'            ! 5: dispersion variable, initial w (ft/sec)
       1.81966226326447
      -1.66065644247974
       1.63242852500904
```

```
      -2.57409847127192
       2.15945300592152
      -5.09542176195067
      0.683309458727747
      -1.75997503233907
      -1.85610150991851
      0.364420135180757
     -0.808423505900293
       2.32982572256186
     -0.257367257783369
       2.43238051019613
      -1.14083221968611
      0.165475548111806
      0.956620962972421
        2.6791829646963
       1.48908270198622
     -0.690418761188058
'XBODYIC(11)'          ! 6: dispersion variable, initial q (rad/sec)
      0.715966761076295
       1.59864791070648
      -2.06474090606851
      -0.74363164216871
      0.176184500993892
      0.527838831614006
     -0.553152628988413
      0.298279843028805
       -1.2266067996961
     -0.189676084637622
     -0.301713308303824
      0.956955953429327
     -0.533365873208898
     -0.901081815072339
     -0.892551536598023
      0.278716514811784
      -0.74580671010209
       1.60346351445097
      0.574269619123481
      0.320654602066792
'XBODYIC(12)'          ! 7: dispersion variable, initial r (rad/sec)
     -0.0728008807496808
     -0.994309848332088
      -0.74735830985414
    -0.0308138815521218
       0.98835462606272
     -0.599016631129926
       1.47664355788457
     -0.813801129646501
      0.645039588313332
      -1.30991937253248
     -0.867425111979492
     -0.474233283350728
      0.222417390154482
       1.87132280901829
      0.110001477655435
      -0.41134087365668
      0.511241723948488
      -1.19911651707093
```

```
-0.0963605845520454
  0.445817318125019
```

Appendix J. DIS.OUT File

The DIS.OUT file outputs the projectile states at the point of impact with the ground. All projectile states are recorded in the same order as listed in appendix I for each case in the dispersion simulation. For instance, if 200 simulations are run, the DIS.OUT file will contain 200 lines, each containing the full projectile state at the point of impact.

INTENTIONALLY LEFT BLANK.

List of Symbols, Abbreviations, and Acronyms

α_{C_i}	*ith* canard aerodynamic angle of attack
ϕ, θ, ψ	Euler roll, pitch, and yaw angles of a projectile
ρ	local air density
γ_{C_i}	*ith* canard sweep angle
δ_{C_i}	*ith* canard pitch angle
ϕ_{C_i}	*ith* canard azimuth angle
A_{T_i}	*ith* rocket motor thrust force amplification factor
$C_{\#}$	projectile aerodynamic coefficients
D	projectile characteristic length (diameter)
DOF	degrees of freedom
IMU	inertial measurement unit
L, M, N	external moments on a projectile expressed in the projectile body axes
m	mass of a projectile
$\dot{m}_{R_i}$	*ith* rocket motor mass flow rate
N_{RX_i}	*ith* rocket motor $\vec{i}_B$ direction cosine
N_{RY_i}	*ith* rocket motor $\vec{j}_B$ direction cosine
N_{RZ_i}	*ith* rocket motor $\vec{k}_B$ direction cosine
PNG	proportional navigation guidance
PRODAS	Projectile Rocket Ordnance Design and Analysis System
p, q, r	components of the angular velocity vector of a projectile expressed in the projectile body reference frame
q_{C_i}	dynamic pressure at the *ith* canard computation point
S_{C_i}	*ith* canard reference area

T_{R_i}	*ith* rocket motor thrust force
T_{C_i}	*ith* canard transformation matrix from the canard reference frame to the parent projectile body axis
u, v, w	translation velocity components of the center of mass of a projectile resolved in the projectile body reference frame
W	magnitude of the projectile weight force
x, y, z	components of the center of mass position vector of a projectile expressed in the inertial reference frame
X, Y, Z	external forces on a projectile expressed in the projectile body axes

NO. OF
COPIES ORGANIZATION

1 (PDF only) DEFENSE TECHNICAL INFORMATION CTR
DTIC OCA
8725 JOHN J KINGMAN RD
STE 0944
FORT BELVOIR VA 22060-6218

1 DIRECTOR
US ARMY RESEARCH LAB
IMNE ALC HRR
2800 POWDER MILL RD
ADELPHI MD 20783-1197

1 DIRECTOR
US ARMY RESEARCH LAB
RDRL CIO LL
2800 POWDER MILL RD
ADELPHI MD 20783-1197

1 DIRECTOR
US ARMY RESEARCH LAB
RDRL CIO MT
2800 POWDER MILL RD
ADELPHI MD 20783-1197

1 DIRECTOR
US ARMY RESEARCH LAB
RDRL D
2800 POWDER MILL RD
ADELPHI MD 20783-1197

INTENTIONALLY LEFT BLANK.